IBM® DB2® Spatial Extender

Guía y consulta del usuario

Versión 7

SC10-3528-01

IBM® DB2® Spatial Extender

Guía y consulta del usuario

Versión 7

SC10-3528-01

Nota

Antes de utilizar esta información y el producto al que da soporte, asegúrese de leer la información general contenida en la sección "Avisos" en la página 363.

Segunda edición (junio de 2001)

Esta publicación es la traducción del original inglés *IBM® DB2® Spatial Extender User's Guide and Reference Version 7*, (SC27-0701-01).

Esta edición corresponde a la Versión 7, Release 2 del producto IBM DB2 Spatial Extender, y a todos los releases y modificaciones subsiguientes hasta que se indique lo contrario en nuevas ediciones.

Contenido

Figuras

Tablas

Acerca de este manual

Esta manual está dividido en dos partes. La primera parte contiene información conceptual sobre DB2 Spatial Extender y explica cómo instalar, configurar, administrar y programar para Spatial Extender en sistemas Windows NT y AIX. La segunda parte consta de información de consulta sobre procedimientos almacenados, geometrías, funciones, mensajes y vistas de catálogos que utiliza con Spatial Extender.

A quién va dirigido este manual

Este manual está destinado a administradores que configuran el entorno espacial y a programadores de aplicaciones que desarrollan aplicaciones con datos espaciales.

Convenios

En este manual se utilizan los siguientes convenios de resaltado:

Letra negrita
Indica mandatos y controles de la interfaz gráfica de usuario (GUI) (por ejemplo, nombres de campos, nombres de carpetas, opciones de menús).

`Letra monoespaciado`
Indica ejemplos de codificación o de texto que escribe el usuario.

Letra cursiva
Indica variables que el usuario debe sustituir por un valor. El tipo cursiva indica también títulos de manuales y enfatiza palabras.

TIPO MAYÚSCULAS
Indica palabras clave de SQL y nombres de objetos (por ejemplo, tablas, vistas y servidores).

Cómo enviar sus comentarios

La información que envía ayuda a IBM a ofrecer información de calidad. Por favor, envíe los comentarios que desee sobre este manual o sobre otra documentación de DB2. Puede utilizar cualquiera de los siguientes métodos para transmitir sus comentarios:

- Envíe sus comentarios desde la Web. Puede acceder al formulario de comentarios del lector en línea de IBM Data Management en la dirección http://www.ibm.com/software/data/rcf

- Envíe sus comentarios por correo electrónico a comments@vnet.ibm.com. Asegúrese de incluir el nombre del producto, el número de versión del producto y el nombre y el número de pieza del manual (si se aplica). Si efectúa comentarios sobre un texto específico, por favor, indique dónde se encuentra el texto (por ejemplo, un capítulo y un título de sección, un número de tabla, un número de página o un título de un tema de ayuda).

Parte 1. Utilización de Spatial Extender

Capítulo 1. Acerca de Spatial Extender

Este capítulo contiene una introducción a Spatial Extender y explica su finalidad y los datos que procesa e indica cómo utilizarlo. Este capítulo concluye con una guía rápida sobre el resto del manual.

La finalidad de Spatial Extender

Spatial Extender sirve para crear un *sistema de información geográfica* (GIS): un grupo de objetos, datos y aplicaciones que le ayudan a generar y a analizar información espacial sobre elementos geográficos. Los *elementos geográficos* son los objetos que forman la superficie de la tierra y los objetos que la ocupan. Forman tanto el entorno natural (como por ejemplo ríos, bosques, montañas y desiertos) como el entorno cultural (ciudades, residencias, edificios de oficinas, monumentos, etc.).

La *información espacial* incluye hechos tales como:

- La ubicación de elementos geográficos con respecto a su entorno (por ejemplo, puntos dentro de una ciudad donde se encuentran hospitales y clínicas o la proximidad de las zonas residenciales de una ciudad con respecto a zonas sísmicas locales)
- Modos en que los elementos geográficos se relacionan entre sí (por ejemplo, información sobre que un determinado sistema de ríos queda enclavado en una determinada región, o sobre que ciertos puentes de dicha región cruzan los afluentes del sistema de ríos)
- Las medidas que se utilizan para uno o más elementos geográficos (por ejemplo, la distancia entre un edificio de oficinas y el límite del terreno o la longitud del perímetro de una reserva de aves)

La información espacial, sola o en combinación con datos procedentes de un sistema tradicional de gestión de bases de datos relacionales (RDBMS), le puede ayudar a diseñar proyectos y a tomar decisiones empresariales o políticas. Por ejemplo, supongamos que el responsable de prestaciones sociales de un distrito tiene que comprobar cuáles de los candidatos y receptores de las prestaciones sociales viven realmente dentro del área a la que se aplican las prestaciones. Spatial Extender puede deducir esta información a partir de la ubicación del área y de las direcciones de los candidatos y de los receptores.

Supongamos ahora que el propietario de una cadena de restaurantes desea comenzar el negocio en ciudades cercanas. Para determinar dónde abrir nuevos restaurantes, el propietario necesita respuestas a preguntas como: ¿En

qué puntos de la ciudad se concentra la clientela que suele frecuentar mis restaurantes? ¿Dónde están las principales carreteras? ¿En qué puntos es más baja la tasa de criminalidad? ¿Dónde se encuentran los restaurantes de la competencia? Spatial Extender puede generar información espacial en representaciones visuales para responder a estas preguntas y el RDBMS subyacente puede generar etiquetas y texto que expliquen las representaciones visuales.

En este manual aparecen muchos otros ejemplos de aplicación de Spatial Extender, especialmente en el "Capítulo 7. Recuperación y análisis de información espacial" en la página 69, en el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73 y en el "Capítulo 14. Funciones espaciales para consultas SQL" en la página 199.

Datos que representan elementos geográficos

Esta sección contiene una visión general de los datos que el usuario puede generar, almacenar y manejar para obtener información espacial. Los temas de esta sección son:

- Cómo los datos representan elementos geográficos
- La naturaleza de los datos espaciales
- Modos de generar datos espaciales

Cómo los datos representan elementos geográficos

En Spatial Extender, un elemento geográfico se puede representar mediante una fila de una tabla o vista, o mediante una parte de dicha fila. Por ejemplo, consideremos dos de los elementos geográficos que se mencionan en el tema "La finalidad de Spatial Extender" en la página 3, edificios de oficinas y residencias. En la Figura 1 en la página 5, cada fila de la tabla BRANCHES representa una sucursal de un banco. Como variación, cada fila de la tabla CUSTOMERS de la Figura 1 en la página 5, considerada como una unidad, representa un cliente del banco. Sin embargo, se puede considerar que parte de cada fila (en concreto, las celdas que contienen la dirección de un cliente) representa la residencia del cliente.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141	A	A

Figura 1. Fila de tabla que representa un elemento geográfico; fila de tabla cuyos datos de direcciones representan un elemento geográfico. La fila de datos de la tabla BRANCHES representa una sucursal de un banco. Las celdas de datos de direcciones de la tabla CUSTOMERS representan la residencia de un cliente. Los nombres y direcciones de ambas tablas son ficticios.

Las tablas de la Figura 1 contienen datos que identifican y describen las sucursales del banco y sus clientes. Dichos datos se denominan *datos de atributo.*

Un subconjunto de los datos de atributo—los valores que representan direcciones de sucursales y de clientes—se pueden convertir en valores que aportan información espacial. Por ejemplo, tal como se muestra en la Figura 1, la dirección de una sucursal es 92467 Airzone Blvd., San Jose CA 95141. La dirección de un cliente es 9 Concourt Circle, San Jose CA 95141. Spatial Extender puede convertir estas direcciones en valores que indiquen dónde están situadas la sucursal y la residencia del cliente con respecto a sus alrededores. La Figura 2 muestra las tablas BRANCHES y CUSTOMERS con nuevas columnas designadas para contener dichos valores.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141		A	A

Figura 2. Tablas con columnas espaciales añadidas. En cada tabla, la columna LOCATION contendrá coordenadas correspondientes a las direcciones.

Cuando se utilizan direcciones e identificadores parecidos como punto inicial para la información espacial, reciben el nombre de *datos fuente.* Puesto que los

valores obtenidos a partir de los mismos aportan información espacial, estos valores obtenidos se denominan *datos espaciales.* La siguiente sección describe los datos espaciales y contiene una introducción a sus tipos de datos asociados.

La naturaleza de los datos espaciales

Muchos datos espaciales están formados por coordenadas. Una *coordenada* es un número que indica una posición relativa a un punto de referencia. Por ejemplo, las latitudes son coordenadas que indican posiciones con respecto al ecuador. Las longitudes son coordenadas que indican posiciones con respecto al meridiano de Greenwich. De este modo, la posición del Parque Nacional de Yellowstone se puede definir por su latitud (44,45 grados al norte del ecuador) y su longitud (110,40 grados al oeste del meridiano de Greenwich).

Las latitudes, las longitudes, sus puntos de referencia y otros parámetros asociados reciben, en conjunto, el nombre de *sistema de coordenadas.* También hay sistemas de coordenadas basados en valores que no son latitud y longitud. Estos sistemas de coordenadas tienen sus propias medidas de posición, puntos de referencia y parámetros distintivos adicionales.

El dato elemental espacial más simple consta de dos coordenadas que definen la posición de un elemento geográfico individual. (Un *dato elemental* es el valor o valores que ocupan una celda de una tabla relacional.) Un dato elemental espacial más extenso consta de varias coordenadas que definen un recorrido lineal, tal como el que formado por un camino o un río. Un tercer tipo consta de coordenadas que definen el perímetro de un zona; por ejemplo, los límites de una parcela de tierra o de la zona que queda inundada tras la crecida de un río. Estos y otros tipos de datos elementales espaciales a los que da soporte Spatial Extender se describen con detalle en el “Capítulo 13. Geometrías y funciones espaciales asociadas” en la página 159.

Cada dato espacial es una instancia de un tipo de datos espaciales. El tipo de datos correspondiente a dos coordenadas que marcan una ubicación es ST_Point; el tipo de datos correspondiente a coordenadas que definen recorridos lineales es ST_LineString; y el tipo de datos correspondiente a coordenadas que definen perímetros es ST_Polygon. Estos tipos, junto con los demás tipos de datos para datos espaciales, son tipos estructurados que pertenecen a una sola jerarquía. Para ver una visión general de la jerarquía, consulte la sección “Acerca de los tipos de datos espaciales” en la página 45.

De dónde proceden los datos espaciales

Puede obtener datos espaciales:

- A partir de datos de atributo
- A partir de otros datos espaciales
- Importándolos

Utilización de datos de atributo como datos fuente

Spatial Extender puede obtener datos espaciales a partir de datos de atributo, tales como direcciones (tal como se menciona en el tema "Cómo los datos representan elementos geográficos" en la página 4). Este proceso se denomina *geocodificación.* Para comprender la secuencia de procesos que intervienen, considere que la Figura 2 en la página 5 es una imagen "anterior" y que la Figura 3 es una imagen "posterior". La Figura 2 en la página 5 muestra que la tabla BRANCHES y la tabla CUSTOMERS tienen una columna que sólo contiene valores nulos para los datos espaciales. Supongamos que Spatial Extender geocodifica los datos de direcciones contenidos en estas tablas para obtener las coordenadas correspondientes a las direcciones y coloca las coordenadas en las columnas. La Figura 3 muestra este resultado.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	1653 3094

CUSTOMERS

ID	LAST NAME	FIRST NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	CHECKING	SAVINGS
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	CA	95141	953 1527	A	A

Figura 3. Tablas que incluyen datos espaciales obtenidos a partir de datos fuente. La columna LOCATION de la tabla CUSTOMERS contiene coordenadas que un geocodificador ha obtenido a partir de la dirección contenida en las columnas ADDRESS, CITY, STATE y ZIP. De forma similar, la columna LOCATION de la tabla BRANCHES contiene coordenadas que el geocodificador ha obtenido a partir de la dirección de las columnas ADDRESS, CITY, STATE y ZIP de esta tabla. Este ejemplo es ficticio; se muestran coordenadas simuladas, no reales.

Spatial Extender utiliza una función, denominada un *geocodificador* , para convertir datos de atributo en datos espaciales y para colocar estos datos espaciales en columnas de tablas. Para obtener más información sobre geocodificadores, consulte la sección "Acerca de la geocodificación" en la página 53.

Utilización de otros datos espaciales como datos fuente

Se pueden generar datos espaciales no sólo a partir de datos de atributo, sino también a partir de otros datos espaciales. Por ejemplo, supongamos que el banco cuyas sucursales están definidas en la tabla BRANCHES desea saber el número de clientes situados a menos de cinco millas de cada sucursal. Para que el banco pueda obtener esta información de la base de datos, debe suministrar a la misma la definición de la zona que queda dentro de un radio de cinco millas alrededor de cada sucursal. Una función de Spatial Extender, ST_Buffer, puede crear esta definición. Utilizando las coordenadas de cada sucursal como entrada, ST_Buffer puede generar las coordenadas que marcan

los perímetros de las zonas deseadas. La Figura 4 muestra la tabla BRANCHES con información suministrada por ST_Buffer.

BRANCHES

ID	NAME	ADDRESS	CITY	STATE	ZIP	LOCATION	SALES_AREA
937	Airzone-Multern	92467 Airzone Blvd	San Jose	CA	95141	1653 3094	1002 2001, 1192 3564, 2502 3415, 1915 3394, 1002 2001

Figura 4. Tabla que incluye nuevos datos espaciales obtenidos a partir de datos espaciales existentes. La función ST_Buffer ha obtenido las coordenadas de la columna SALES_AREA a partir de las coordenadas de la columna LOCATION. Al igual que las coordenadas de la columna LOCATION, las de la columna SALES_AREA son también simuladas; no son reales.

Además de ST_Buffer, Spatial Extender ofrece otras funciones para obtener nuevos datos espaciales a partir de datos espaciales existentes. Para ver descripciones de ST_Buffer y de estas otras funciones, consulte la sección "Funciones que generan geometrías nuevas a partir de geometrías existentes" en la página 188.

Importación de datos espaciales

Un tercer modo de obtener datos espaciales consiste en importarlos de archivos que estén en uno de los formatos a los que da soporte Spatial Extender. Para ver descripciones de estos formatos, consulte el "Capítulo 16. Formatos de archivo para datos espaciales" en la página 337. Estos archivos contienen datos que se suelen aplicar a correlaciones: seguimientos del censo, zonas que quedan inundadas tras la crecida de un río, fallas, etc. Al utilizar dichos datos en combinación con datos espaciales generados por el usuario, puede aumentar la información geográfica disponible. Por ejemplo, si un departamento de obras públicas tuviera que determinar a qué peligros se expone una comunidad residencial, podría utilizar ST_Buffer para definir una zona alrededor de la comunidad. El departamento de obras públicas podría importar datos sobre zonas que quedan inundadas tras la crecida de un río o fallas para ver cuáles de estas áreas de problemas se aplican a la zona.

Cómo crear y utilizar un GIS de Spatial Extender

Para crear un GIS de Spatial Extender debe configurar Spatial Extender y desarrollar proyectos GIS dentro de los entornos combinados de Spatial Extender y su RDBMS de DB2 asociado. Puede utilizar el GIS aplicando estos proyectos; es decir, generando y analizando la información (espacial o convencional) que proporcionan. Para ello debe llevar a cabo varios grupos de tareas. Esta sección contiene una introducción a las interfaces con las que puede realizar estas tareas, ofrece una visión general de las tareas y presenta un ejemplo que las ilustra.

Interfaces con Spatial Extender y funciones asociadas

Esta sección describe las interfaces mediante las que puede crear un GIS de Spatial Extender (es decir, definir recursos para él, obtener datos espaciales, etc.) y utilizarlo (es decir, generar y analizar información sobre elementos geográficos).

Puede crear un GIS de Spatial Extender:

- Utilizando las opciones de menús y ventanas de Spatial Extender del Centro de Control de DB2. Para obtener instrucciones, consulte:
  - "Capítulo 3. Preparación de recursos" en la página 33
  - "Capítulo 4. Cómo definir columnas espaciales, registrarlas como capas y habilitar un geocodificador para que las mantenga" en la página 45
  - "Capítulo 5. Cómo llenar columnas espaciales" en la página 53
  - "Capítulo 6. Creación de índices espaciales" en la página 67
- Ejecutando un programa de aplicación que llame a procedimientos almacenados de Spatial Extender. Para obtener directrices sobre cómo desarrollar un programa de este tipo, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.
- Utilizando el Centro de Control y un programa de aplicación. Por ejemplo, puede utilizar el Centro de Control para invocar el geocodificador por omisión. Si, además, desea utilizar otro geocodificador, primero debe registrarlo ante Spatial Extender invocando el procedimiento almacenado db2gse.gse_register_gc en un programa de aplicación. (Para obtener información sobre geocodificadores que no son el geocodificador por omisión, consulte la sección "Acerca de la geocodificación" en la página 53. Para obtener información sobre el procedimiento almacenado db2gse.gse_register_gc, consulte la sección "db2gse.gse_register_gc" en la página 108.)
- Utilizando el Centro de Control, un programa de aplicación, o ambos junto con otras interfaces. Por ejemplo, para crear una tabla para albergar datos generados por una función espacial, tal como un geocodificador, podría utilizar las interfaces del Procesador de Línea de Mandatos o del Centro de Control.

Puede utilizar un GIS de Spatial Extender mediante:

- La representación gráfica de información con un geoexplorador; por ejemplo ArcExplorer para Java, Versión 3.0, proporcionado por ESRI (Environmental Systems Research Institute)
- El envío de consultas de SQL de forma explícita desde el Centro de Control o el procesador de línea de mandatos de DB2
- El envío de consultas de SQL desde un programa de aplicación

Tareas para crear y utilizar un GIS de Spatial Extender

Esta sección contiene una visión general de las tareas a realizar para crear y utilizar un GIS de Spatial Extender. Las tareas a realizar para crear el GIS incluyen la configuración de Spatial Extender y el desarrollo de proyectos GIS. Las tareas a realizar para utilizar el GIS incluyen la implantación de los proyectos. Esta visión general empieza por la configuración de Spatial Extender y luego describe el desarrollo e implantación de un proyecto GIS. Esta sección concluye indicando cómo las tareas descritas en esta visión general pueden variar en la práctica real.

Configuración de Spatial Extender

Para configurar Spatial Extender:

1. Planifique y prepare (decida qué proyectos GIS desea desarrollar y qué base de datos debe habilitar para Spatial Extender, seleccione personal para administrar Spatial Extender y desarrollar los proyectos, etc).
2. Instale Spatial Extender.
3. Prepare recursos para dar soporte a los proyectos GIS; por ejemplo:

   Recursos suministrados por Spatial Extender
   Estos incluyen un catálogo del sistema, tipos de datos espaciales, funciones espaciales (incluido un geocodificador por omisión), etc. La tarea de preparar estos recursos se denomina *habilitación de la base de datos para operaciones espaciales.*

   Geocodificadores desarrollados por usuarios, proveedores o ambos
   El geocodificador por omisión convierte direcciones de Estados Unidos en datos espaciales. Su organización y otras pueden suministrar geocodificadores que conviertan direcciones de otros países y otros tipos de datos de atributo en datos espaciales.

Para obtener instrucciones sobre cómo instalar Spatial Extender, consulte el "Capítulo 2. Instalación de Spatial Extender" en la página 17. Para obtener instrucciones sobre cómo utilizar el Centro de Control para preparar recursos, consulte el "Capítulo 3. Preparación de recursos" en la página 33. Para ver directrices sobre cómo utilizar un programa de aplicación para esta finalidad, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73. Para ver un ejemplo que ilustre la tarea de configurar Spatial Extender, consulte la sección "Un sistema para integrar datos espaciales y tradicionales" en la página 13.

Desarrollo y aplicación de un proyecto GIS

Para desarrollar y aplicar un proyecto GIS:

1. Planifique y prepare (defina objetivos del proyecto, decida qué tablas y datos necesita, determine qué sistema o sistemas de coordenadas utilizar, etc.)

2. Decida qué sistema o sistemas de referencia espacial utilizar. Los valores de las coordenadas suelen incluir enteros positivos, números negativos y números decimales. Sin embargo, Spatial Extender debe almacenar todos los valores de coordenadas en formato de enteros positivos. Un *sistema de referencias espaciales* es un grupo de parámetros que define el modo en que los números negativos y decimales se deben convertir en enteros positivos para que Spatial Extender los pueda almacenar. Después de decidir qué sistema de coordenadas utilizar para una columna espacial, tiene que especificar el sistema de referencias espaciales según el que se llevará a cabo la conversión necesaria correspondiente a dicha columna. Si un sistema de referencias espaciales se ajusta a sus requisitos, puede utilizarlo; si no es así, puede crear uno.
3. Defina una o más columnas para que contengan datos espaciales, regístrelas ante Spatial Extender y habilite un geocodificador para mantenerlas de forma automática.

   Para registrar una columna espacial hay que registrarla en el catálogo de Spatial Extender. Una vez registrada se denomina *capa*, porque la información generada a partir de ella añadirá un estrato, o capa, al paisaje geográfico virtual generado por GIS. Una vez registrada la columna, puede realizar operaciones espaciales sobre ella; por ejemplo, puede llenarla con datos y definir un índice espacial en ella.
4. Llene con datos columnas espaciales:
   - Para un proyecto que necesite un geocodificador, defina parámetros para el geocodificador. Luego, ejecute el geocodificador de modo que, en una sola operación, geocodifique todos los datos fuente disponibles y cargue las coordenadas resultantes en una capa.
   - Para un proyecto que necesite que se importen datos espaciales, importe los datos.
5. Facilite acceso a columnas espaciales. En concreto, esto implica definir índices que permitan a DB2 acceder a datos espaciales de forma rápida y definir vistas que permitan a los usuarios recuperar datos interrelacionados de forma eficiente. Después de definir una vista de este tipo, tiene que registrar sus columnas espaciales como capas.
6. Genere y analice información espacial e información comercial relacionada. Esto implica consultar columnas espaciales y columnas de atributo asociadas. En dichas consultas, incluirá funciones de Spatial Extender que devuelven una gran variedad de información; por ejemplo, la distancia mínima entre dos elementos geográficos, o las coordenadas que definen una zona que circunda a un elemento geográfico. Para obtener información sobre la función que devuelve dichas coordenadas, ST_Buffer, consulte la sección "Utilización de otros datos espaciales como datos fuente" en la página 7 y el tema "ST_Buffer" en la página 231. Para ver ejemplos de consultas que utilizan funciones espaciales, consulte el

"Capítulo 7. Recuperación y análisis de información espacial" en la página 69 y el "Capítulo 14. Funciones espaciales para consultas SQL" en la página 199.

Para obtener instrucciones sobre cómo utilizar el Centro de Control para llevar a cabo tareas relacionadas con el desarrollo de un proyecto GIS, consulte:

- "Capítulo 3. Preparación de recursos" en la página 33
- "Capítulo 4. Cómo definir columnas espaciales, registrarlas como capas y habilitar un geocodificador para que las mantenga" en la página 45
- "Capítulo 5. Cómo llenar columnas espaciales" en la página 53
- "Capítulo 6. Creación de índices espaciales" en la página 67

Para obtener instrucciones sobre cómo utilizar el Centro de Control para implantar un proyecto GIS, consulte el "Capítulo 7. Recuperación y análisis de información espacial" en la página 69.

Para obtener instrucciones sobre cómo utilizar un programa de aplicación para desarrollar e implantar un proyecto GIS, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Para ver un ejemplo que muestra la tarea en su conjunto, consulte "Un proyecto para establecer sucursales y ajustar primas" en la página 13.

Cómo pueden variar los grupos de tareas

El grupo de tareas que puede llevar a cabo para crear y utilizar un GIS de Spatial Extender puede variar en contenido y secuencia, en función de sus necesidades y de las interfaces que utilice. Por ejemplo, considere las tareas de definir columnas que contienen datos espaciales, registrarlas como capas y habilitar un geocodificador para que las mantenga de forma automática. Con el Centro de Control, puede llevar a cabo estas tareas en conjunto, desde una sola ventana. Sin embargo, si invoca procedimientos almacenados desde un programa, puede realizar estas tareas por separado y las puede planificar como desee.

Ejemplo: una compañía de seguros actualiza su GIS

Esta sección presenta un caso práctico para ilustrar los grupos de tareas descritos en la sección anterior.

El entorno de sistemas de información de la compañía de seguros Safe Harbor Real Estate incluye un sistema DB2 Universal Database y un sistema de gestión de bases de datos GIS separado. Hasta cierto punto, las consultas pueden recuperar combinaciones de datos procedentes de los dos sistemas. Por ejemplo, una tabla de DB2 almacena información sobre ingresos y una tabla de GIS almacena las ubicaciones de las sucursales de la compañía. Por lo

tanto, es posible encontrar las ubicaciones de las sucursales que aportan ingresos de determinadas cantidades. Pero los datos procedentes de los dos sistemas no se pueden integrar (por ejemplo, los usuarios no pueden unir columnas de DB2 con columnas de GIS) y los servicios de DB2, como la optimización de consultas, no están disponibles para GIS. Para solucionar estos problemas, Safe Harbor adquiere Spatial Extender y establece un nuevo departamento de desarrollo de GIS. Las siguientes secciones describen cómo el departamento configura Spatial Extender y lleva a cabo su primer proyecto.

Un sistema para integrar datos espaciales y tradicionales

Para configurar Spatial Extender, el departamento de desarrollo de GIS de Safe Harbor actúa del siguiente modo:

1. El departamento se prepara para incluir Spatial Extender en su entorno DB2. Por ejemplo:
   a. El equipo de dirección del departamento asigna a un equipo de administración espacial la tarea de instalar e implantar Spatial Extender y a un equipo de análisis espacial la tarea de generar y analizar información espacial.
   b. Puesto que las decisiones comerciales de Safe Harbor se basan principalmente en los requisitos de los clientes, el equipo de dirección decide instalar Spatial Extender en la base de datos que contiene información sobre los clientes. La mayor parte de esta información se almacena en una tabla denominada CUSTOMERS.

      Para hacer referencia fácilmente a la base de datos seleccionada, los miembros del departamento de desarrollo de GIS la llaman *base de datos GIS*. Sin embargo, saben que no está reservada únicamente para proyectos GIS; las aplicaciones no espaciales la pueden seguir utilizando, como antes.
2. El equipo de administración espacial instala Spatial Extender.
3. El equipo de administración espacial prepara los recursos que serán necesarios para los proyectos GIS:
   - El equipo utiliza el Centro de Control para suministrar los recursos que habilitan la base de datos GIS para operaciones espaciales. Estos recursos incluyen el catálogo de Spatial Extender, tipos de datos espaciales, funciones espaciales, etc.
   - Puesto que Safe Harbor está empezando a realizar operaciones comerciales en Canadá, el equipo de administración espacial empieza solicitando a los proveedores canadienses geocodificadores para convertir las direcciones de Canadá en datos espaciales.

Un proyecto para establecer sucursales y ajustar primas

Para llevar a cabo su primer proyecto GIS bajo Spatial Extender, el equipo de desarrollo de GIS actúa del siguiente modo:

1. El departamento se prepara para desarrollar el proyecto; por ejemplo:

- El equipo de dirección define objetivos para el proyecto:
  - Determinar dónde establecer nuevas sucursales.
  - Fijar primas de riesgo en función de la proximidad de los clientes a zonas de riesgo (zonas con alto nivel de accidentes de tráfico, zonas con alto nivel de criminalidad, zonas que se inundan tras la crecida de un río, zonas sísmicas, etc.)
- El proyecto GIS tratará con clientes y sucursales de Estados Unidos. Por lo tanto, el equipo de administración espacial decide:
  - Utilizar sistemas de coordenadas que definan con precisión las ubicaciones en las zonas de Estados Unidos en las que Safe Harbor realiza operaciones comerciales.
  - Utilizar el geocodificador por omisión, porque está diseñado para geocodificar direcciones de Estados Unidos.
- El equipo de administración espacial decide qué datos se necesitan para cumplir con los objetivos del proyecto y qué tablas contendrán dichos datos.

2. Mediante el Centro de Control, el equipo de administración espacial crea dos sistemas de referencias espaciales. Uno determina el modo en que las coordenadas que definen las ubicaciones de las sucursales se deben convertir en elementos de datos que Spatial Extender pueda almacenar. El otro determina el modo en que las coordenadas que definen las ubicaciones de las residencias de los clientes se deben convertir en elementos de datos que Spatial Extender pueda almacenar.
3. Mediante el Centro de Control, el equipo de administración espacial define columnas que contengan datos espaciales, las registra como capas y habilita un geocodificador para que las mantenga de forma automática:
   - El equipo añade una columna LOCATION a la tabla CUSTOMERS. La tabla ya contiene las direcciones de los clientes. El geocodificador por omisión las convertirá en datos espaciales y cargará estos datos en la columna LOCATION.
   - El equipo genera una tabla OFFICES para que contenga los datos que ahora se almacenan en el otro GIS. Estos datos incluyen las direcciones de las sucursales de Safe Harbor, datos espaciales que el geocodificador ha obtenido a partir de estas direcciones y datos espaciales que definen una zona dentro de un radio de cinco millas alrededor de cada sucursal. Los datos generados por el geocodificador irán en una columna LOCATION. Los datos que definen las zonas irán en una columna SALES_AREA.
   - El equipo registra las dos columnas LOCATION y las columnas SALES_AREA como capas.
   - El equipo habilita el geocodificador por omisión para que mantenga de forma automática las dos columnas LOCATION.

4. El equipo de administración espacial llena la columna LOCATION de la tabla CUSTOMER, toda la tabla OFFICES y una nueva tabla HAZARD_ZONES:
   - El equipo utiliza el Centro de Control para llenar la columna LOCATION de la tabla CUSTOMER:
     a. El equipo indica al geocodificador que inserte datos espaciales correspondientes a una dirección en la columna LOCATION únicamente bajo la siguiente condición: la coincidencia entre la dirección y su equivalente en los registros de la Oficina de censo de Estados Unidos debe ser del 100 por ciento. (Con Spatial Extender se proporciona un archivo de direcciones suministrado por la Oficina de censo. Para que el geocodificador pueda convertir una dirección en datos fuente en datos espaciales, el geocodificador debe intentar hacer coincidir esta dirección con su equivalente en el archivo. Los usuarios especifican el grado de precisión de la coincidencia para que los datos espaciales se coloquen en una tabla. Este porcentaje se denomina *precisión*.)
     b. El equipo ejecuta el geocodificador en modalidad de proceso por lotes, de modo que pueda geocodificar todas las direcciones de la tabla en una operación. Para consternación del equipo, el geocodificador rechaza aproximadamente una de cada diez direcciones.
     c. El equipo supone que los rechazos corresponden a las nuevas direcciones que no tienen coincidencias exactas en los registros de la Oficina de censo. Para resolver el problema, el equipo reduce la precisión a 85.
     d. El equipo vuelve a ejecutar el geocodificador en modalidad de proceso por lotes. La tasa de direcciones rechazadas desciende a un nivel aceptable.
   - Mediante el programa de utilidad que se suministra por el GIS que va por separado, el equipo carga los datos de sucursales en un archivo. Luego el equipo utiliza el Centro de Control para importar estos datos del archivo a la nueva tabla OFFICES.
   - Mediante el Centro de Control, el equipo crea una tabla HAZARD ZONES, registra sus columnas espaciales como capas e importa datos en la misma. Los datos proceden de un archivo suministrado por un proveedor de mapas.
5. Mediante el Centro de Control, el equipo de administración espacial facilita acceso a las nuevas capas:
   - El equipo crea índices para las mismas.
   - El equipo crea una vista que une columnas procedentes de las tablas CUSTOMERS y HAZARD ZONES. Luego el equipo registra las columnas espaciales de la vista como capas.

6. El equipo de análisis espacial ejecuta consultas para obtener información que ayudará a cumplir con los objetivos originales: determinar dónde establecer nuevas sucursales y ajustar las primas en función de la proximidad de los clientes a zonas de riesgo.

Capítulo 2. Instalación de Spatial Extender

Este capítulo proporciona instrucciones para instalar DB2 Spatial Extender para AIX, Windows NT y Windows 2000. También se tratan los temas siguientes:

- Configuración de DB2 Spatial Extender
- Requisitos del sistema
- Instalación de DB2 Spatial Extender para Windows NT y Windows 2000
- Instalación de DB2 Spatial Extender para AIX
- Verificación de la instalación
- Consideraciones posteriores a la instalación
- Invocación de Spatial Extender

Configuración de DB2 Spatial Extender

Un sistema Spatial Extender consta de DB2 Universal Database, Spatial Extender y un geoexplorador (tal como, ArcExplorer para Java, Versión 3.0). Normalmente, hay una base de datos habilitada para operaciones espaciales en el servidor. Puede utilizar aplicaciones clientes para acceder a datos espaciales a través de consultas espaciales y procedimientos almacenados de Spatial Extender. Puede también configurar DB2 Spatial Extender en un entorno autónomo, que es una configuración en la que el cliente y el servidor residen en la misma máquina. Tanto en la configuración cliente-servidor como en la configuración autónoma, puede visualizar datos espaciales utilizando un geoexplorador.

IBM no proporciona actualmente un geoexplorador que pueda mostrar resultados visuales procedentes de consultas. Para obtener más información sobre los geoexploradores, vea la sección “Cómo bajar ArcExplorer” en la página 29.

La Figura 5 ilustra la arquitectura de Spatial Extender.

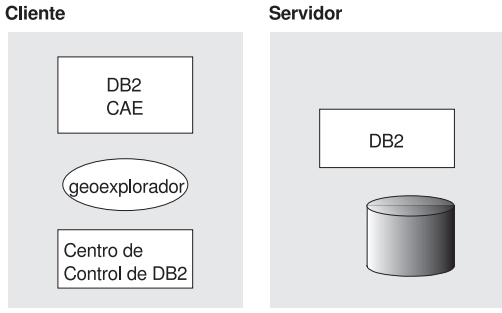


Figura 5. Configuración de cliente-servidor

Requisitos del sistema

Esta sección explica los requisitos de software y hardware de DB2 Spatial Extender.

Sistemas operativos soportados

Spatial Extender se puede instalar en AIX Versión 4.2 (o posterior), Windows NT 4.0 (o posterior) con el Service Pack 5, y Windows 2000.

Nota: Si desea utilizar ArcSDE para representar y ver datos espaciales, es necesario AIX Versión 4.3.3 o posterior.

Software de base de datos necesario

Antes de instalar DB2 Spatial Extender, debe tener instalado y configurado el siguiente software de DB2 en el cliente y en el servidor:

Software de cliente

Para los productos de cliente de DB2 Spatial Extender:

- DB2 Administration Client, Versión 7.1

  Si no tiene intención de utilizar el Centro de Control de DB2, un geoexplorador para acceder a datos espaciales ni el programa de ejemplo de DB2 Spatial Extender, puede instalar y utilizar DB2 Administration Client, Versión 6.0.
- Fixpack 1

DB2 Administration Client junto con el Fixpack 1 se instalan automáticamente en el sistema al instalar el cliente de DB2 Spatial Extender desde el CD-ROM.

Importante: Si en el cliente está instalado DB2 Universal Database Enterprise Edition, Versión 7.1, o DB2 Universal Database Enterprise-Extended Edition, Versión 7.1, debe instalar el Fixpack 1 *antes* de instalar DB2 Spatial Extender.

Software de servidor

Para los productos de servidor de DB2 Spatial Extender, es necesario instalar uno de los siguientes productos de servidor en el sistema *antes* de instalar DB2 Spatial Extender:

- DB2 Universal Database Enterprise Edition, Versión 7.1, con el Fixpack 1
- DB2 Universal Database Enterprise-Extended Edition, Versión 7.1, con el Fixpack 1

Si desea utilizar el Centro de Control de DB2, debe crear y configurar un servidor DAS (DB2 Administration Server). Para obtener más información sobre la creación y configuración del servidor DAS, vea el manual *IBM DB2 Universal Database Administration Guide: Implementation*.

Nota: Aunque puede utilizar DB2 Spatial Extender con DB2 Universal Database Enterprise-Extended Edition, el índice espacial no se puede particionar entre varios nodos como ocurre en el entorno de proceso masivo en paralelo (MPP).

Requisitos de espacio de disco

La Tabla 1 muestra los requisitos de espacio de disco correspondientes a Spatial Extender.

Tabla 1. Requisitos de espacio de disco

Productos de Spatial Extender	Espacio de disco
Productos de servidor de DB2 Spatial Extender: • Código de biblioteca del servidor de Spatial Extender, datos de referencia de ejemplo para el geocodificador y documentación • Elemento opcional y disponible en un CD-ROM independiente: datos de referencia para geocodificador (Estados Unidos) Para obtener más información sobre la utilización de los datos de referencia del geocodificador, vea "Datos de referencia del geocodificador de DB2 Spatial Extender" en la página 30.	594 MB de espacio total de disco: • 31 MB (código de biblioteca del servidor de Spatial Extender, datos de referencia de ejemplo para el geocodificador y documentación) • 563 MB (datos de referencia del geocodificador para Estados Unidos)
Productos de cliente de DB2 Spatial Extender (incluye los datos del programa de ejemplo)	1 MB

Tabla 1. Requisitos de espacio de disco (continuación)

Productos de Spatial Extender	Espacio de disco
	Nota: El espacio de disco necesario calculado en esta tabla presupone que se ha instalado DB2 Universal Database y DB2 Spatial Extender con un tipo de instalación estándar en un sistema Windows NT o Windows 2000, o en un sistema AIX con componentes preseleccionados. Si instala DB2 Spatial Extender o ha instalado DB2 Universal Database con otro tipo de instalación, los cálculos del espacio de disco pueden variar.

Instalación de DB2 Spatial Extender para Windows NT y Windows 2000

Para instalar Spatial Extender para Windows NT y Windows 2000:

1. Inserte el CD-ROM de Spatial Extender en la unidad de CD-ROM. Se abrirá el Panel de inicio de aplicaciones de DB2, que es una interfaz desde la que puede instalar DB2 Spatial Extender.
2. Pulse **Instalar**. Después de la inicialización del programa, se abre la ventana Seleccionar productos.

   Notas:

   a. En cualquier momento durante la instalación, puede pulsar **Cancelar** para interrumpir la instalación y salir de ella.
   b. Si recibe un mensaje de aviso que indica que DB2 está en ejecución y bloqueado por una lista de procesos, pulse **Sí** *sólo* si la base de datos no se está utilizando y no hay usuarios importantes conectados. DB2 concluirá esos procesos y no intentará guardar datos. Si está instalando en un sistema activo, utilice otros métodos administrativos para concluir los procesos bloqueados de DB2.
   c. Si está instalando DB2 Spatial Extender en un entorno autónomo, seleccione **DB2 Administration Client** en el programa de instalación de Windows NT y Windows 2000. La opción **DB2 Administration Client *no*** está preseleccionada en el programa de instalación.
3. Seleccione en la lista los productos que desea instalar:
   - Seleccione **DB2 Spatial Extender Server** y **DB2 Administration Client** si está instalando DB2 Spatial Extender en una configuración autónoma.

- Seleccione **DB2 Spatial Extender Server** si está instalando DB2 Spatial Extender en una plataforma de servidor.
- Seleccione **DB2 Administration Client** si está instalando DB2 Spatial Extender en una plataforma de cliente.

La opción **DB2 Administration Client** instala subcomponentes de DB2 Universal Database, los datos y programa de ejemplo de Spatial Extender y el Centro de Control de DB2 Universal Database. La opción **DB2 Spatial Extender Server** comprende el código de biblioteca del servidor de DB2 Spatial Extender, datos de referencia de ejemplo para el geocodificador y documentación.

Nota: El producto de servidor de DB2 Spatial Extender comprende ***sólo*** una parte de los datos existentes del geocodificador. El conjunto completo de datos de referencia del geocodificador se proporciona en un CD-ROM independiente, que se entrega junto con DB2 Spatial Extender.

4. Pulse **Siguiente**. Se abrirá la ventana Seleccionar tipo de instalación.
5. Seleccione el tipo de instalación. Si selecciona **Personalizada**, se abre la ventana Seleccionar componentes. Puede seleccionar los componentes que desea instalar. Para ello, debe tener conocimientos sobre los componentes de DB2 y sus valores.
6. Pulse **Siguiente** para abrir la ventana Elegir ubicación de destino.
7. Elija la carpeta donde desea instalar DB2 Spatial Extender. Pulse **Examinar** si desea elegir una carpeta que no sea la carpeta por omisión.

   Nota: Si DB2 Universal Database ya está instalado en el sistema, no puede elegir una nueva ubicación ni crear una nueva carpeta desde esta ventana. La ventana Elegir ubicación de destino muestra lo siguiente:
   - La ubicación de la carpeta donde se instalará DB2 Spatial Extender
   - El espacio de disco necesario para instalar DB2 Spatial Extender
8. Pulse **Siguiente** para instalar DB2 Spatial Extender. Se abrirá la ventana Progreso de la instalación, la cual muestra la marcha del procedimiento de instalación.

Instalación de DB2 Spatial Extender para AIX

Esta sección describe los pasos que debe realizar para instalar DB2 Spatial Extender para AIX. Puede utilizar el programa de instalación de DB2 (de forma interactiva o desatendida), la herramienta SMIT (System Management Interface Tool) o el mandato **installp**. Esta sección trata los temas siguientes:

- Montaje del CD-ROM

- Utilización del programa de instalación de DB2
- Utilización de SMIT y del mandato **installp**
- Definición del entorno de instancia de DB2 Spatial Extender

Montaje del CD-ROM

En varios pasos de la instalación de DB2 Spatial Extender para AIX, debe montar el CD-ROM. Cada vez que aparezca el patrón de caracteres */cdrom* en el presente documento, debe realizar el procedimiento de montaje descrito en esta sección *antes* de iniciar el paso en cuestión. Las instrucciones de montaje siguientes son aplicables a instalaciones realizadas mediante el programa de instalación de DB2, SMIT o el mandato **installp**.

Para montar el CD-ROM:

1. Inicie la sesión como usuario con autorización root.
2. Inserte el CD-ROM en la unidad.
3. Emita el mandato siguiente para crear un directorio en el que montar el CD-ROM:

   ```
   mkdir -p /cdrom
   ```

   donde */cdrom* es el directorio de montaje del CD-ROM.
4. Asigne un sistema de archivos de CD-ROM entrando el mandato siguiente:

   ```
   smitty storage
   ```
5. Seleccione **Sistema de archivos**.
6. Seleccione **Añadir/cambiar/mostrar/suprimir sistema de archivos**.
7. Seleccione **Sistema de archivos de CD-ROM**.
8. Seleccione **Añadir sistema de archivos de CD-ROM**.
9. Seleccione **Nombre de dispositivo**.

   Los nombres de dispositivo para sistemas de archivos de CD-ROM deben ser exclusivos. Si el nombre de un sistema existente es igual al que desea seleccionar, suprima ese sistema de archivos de CD-ROM existente o utilice otro nombre para el directorio.
10. En la ventana emergente, escriba el punto de montaje siguiente:

    ```
    /cdrom
    ```
11. Monte el sistema de archivos de CD-ROM entrando el mandato siguiente:

    ```
    smit mountfs
    ```
12. Escriba el nombre del sistema de archivos (por ejemplo, `/dev/cd0`).
13. Escriba el nombre del directorio (por ejemplo, `cdrom`).
14. Escriba el tipo de sistema de archivos (por ejemplo, `cdrfs`).
15. Seleccione **Sí** para la opción Montar sistema de SÓLO LECTURA.
16. Finalice la sesión.

Utilización del programa de instalación de DB2

Esta sección describe los temas siguientes:

- Instalación interactiva
- Instalación desatendida
- Generación de un archivo de anotaciones de rastreo

Durante una instalación en AIX utilizando el programa de instalación de DB2, puede elegir instalar interactivamente o en un entorno desatendido. En la instalación interactiva, el usuario interactúa con una serie de pantallas para instalar y configurar DB2 Spatial Extender. En la instalación desatendida, el usuario debe proporcionar la información de instalación y configuración en un archivo de respuestas, que el usuario crea antes de invocar el programa de instalación de DB2. Si necesita instalar DB2 Spatial Extender en varias máquinas, puede personalizar el archivo y utilizarlo para instalar el producto en varias estaciones de trabajo.

Para instalar DB2 Spatial Extender interactivamente:

1. Inicie la sesión como usuario root a la máquina cliente o servidor.
2. Inserte el CD-ROM en la unidad de CD-ROM.
3. Escriba `cd /cdrom`.
4. Escriba `./db2setup`. Se iniciará el programa de instalación de DB2.
5. Seleccione los productos que desea instalar:
   - Seleccione **DB2 Spatial Extender Server** y **Spatial Extender Client** si está instalando DB2 Spatial Extender en una configuración autónoma.
   - Seleccione **DB2 Spatial Extender Server** si está instalando DB2 Spatial Extender en una plataforma de servidor.
   - Seleccione **DB2 Spatial Extender Client** si está instalando DB2 Spatial Extender en un entorno de cliente.
   - Seleccione **DB2 Administration Client** si desea soporte adicional para clientes DB2.

   Utilice la tecla de tabulación para desplazarse hasta el componente que desea seleccionar y pulse Intro. Para cada producto seleccionado existe una ventana de personalización.
6. Seleccione el idioma para los componentes seleccionados.
7. Mediante el tabulador, coloque el cursor en el botón **Aceptar** y pulse Intro para instalar DB2 Spatial Extender.

Para instalar DB2 Spatial Extender de forma desatendida:

1. Cree un archivo de respuestas para una instalación desatendida.
2. Inicie una instalación desatendida.

Estos pasos se describen con detalle en esta sección.

Paso 1. Cree un archivo de respuestas para una instalación desatendida.

Nota: Puede saltarse el Paso 1 y seguir en el Paso 2 si acepta los valores por omisión del archivo de respuestas de ejemplo.

1. Abra el archivo de respuestas de ejemplo para los productos que desea instalar. Los archivos de respuestas de ejemplo están situados en *cdrom*/db2/install/samples, donde *cdrom* es la ubicación de la versión instalable de DB2 Spatial Extender. Existen dos archivos de respuestas de ejemplo: uno para el servidor de Spatial Extender (db2gse.rsp) y otro para el cliente de Spatial Extender (db2gsec.rsp). Los archivos de respuestas de ejemplo contienen:
   - Palabras clave exclusivas de la instalación
   - Valores del Registro correspondientes a variables de entorno
   - Parámetros de configuración del gestor de bases de datos
2. Para cambiar un valor del archivo de respuestas, active una entrada. Para activar una entrada:
   a. Elimine el asterisco (*) que hay a la izquierda de la variable de teclado/entorno.
   b. Borre el valor actual situado a la derecha de la entrada.
   c. Escriba un nuevo valor.
   d. Si realiza cambios, guarde el archivo con otro nombre de archivo para conservar el archivo de respuestas de ejemplo original. Si está instalando directamente desde el CD-ROM, debe guardar en un sistema de archivos local el archivo de respuestas renombrado.

Paso 2. Inicie una instalación desatendida con un archivo de respuestas.

1. Inicie la sesión como usuario con autorización root.
2. Entre el mandato **db2setup**:

   ```
   /cdrom/db2setup -r directorio_archivo_respuestas/archivo_respuestas
   ```

   donde */cdrom* es la ubicación de la imagen instalable de DB2 Spatial Extender, *directorio_archivo_respuestas* es el directorio donde reside el archivo de respuestas y *archivo_respuestas* es el nombre del archivo de respuestas.
3. Compruebe los mensajes del archivo de anotaciones cuando finalice la instalación. La ubicación por omisión del archivo de anotaciones es /tmp/db2setup/db2setup.log.

Generación de un archivo de anotaciones de rastreo para el programa de instalación de DB2: Si tiene problemas con el programa de instalación de DB2, puede crear un archivo de anotaciones de rastreo (db2setup.trc). Puede

enviar el archivo de anotaciones de rastreo y el archivo db2setup.log al Centro de Soporte de IBM para realizar más tareas de diagnóstico. Una vez creados, ambos archivos están situados en el directorio /tmp.

Para generar un archivo de anotaciones de rastreo, ejecute el mandato **db2setup** utilizando el distintivo **-d**, de esta manera: `./db2setup -d`. Esto hace que el programa de instalación de DB2 se ejecute en la modalidad de rastreo. Reproduzca el problema que encontró. Cuando finalice, se creará el archivo de anotaciones de rastreo /tmp/db2setup/db2setup.trc.

Utilización de SMIT o del mandato installp

Para instalar DB2 Spatial Extender utilizando la herramienta SMIT (System Management Interface Tool) o el mandato **installp**:

1. Inicie la sesión como usuario root a la máquina cliente o servidor.
2. Inserte el CD-ROM en la unidad de CD-ROM.
3. Instale DB2 Spatial Extender ejecutando la herramienta SMIT o el mandato **installp**.

   Para ejecutar SMIT:

   a. Entre el mandato **smit install_latest**. Se abrirá el menú de la herramienta SMIT.
   b. Escriba `/cdrom/db2` en el campo Dispositivo/directorio de entrada.
   c. Pulse **EJECUTAR** o pulse Intro para verificar que existe el directorio de instalación.
   d. En el campo **Software a instalar**, especifique si deben instalarse los componentes de cliente o de servidor.

      Para el cliente, escriba: `db2_07_01.gcln`.

      Para el servidor, escriba: `db2_07_01.gsrv`.
   e. Pulse **EJECUTAR** o la tecla Intro (se le solicitará que confirme los parámetros de instalación).
   f. Pulse Intro para confirmar.

      Los archivos del producto se instalarán desde el CD-ROM a la unidad de disco duro, operación que puede tardar unos minutos.
   g. Finalice la sesión.

Definición del entorno de instancia de DB2 Spatial Extender

El mandato **db2icrt** se utiliza para crear nuevas instancias de DB2. Todas las nuevas instancias de DB2 que cree después de instalar DB2 Spatial Extender incluirán DB2 Spatial Extender en el entorno de instancia.

Las instancias de DB2 creadas antes de instalar Spatial Extender no incluyen DB2 Spatial Extender en sus entornos de instancia. Para actualizar instancias de DB2 existentes con Spatial Extender, utilice el mandato **db2iupdt**.

Como usuario con autorización root, escriba el mandato siguiente:

```
db2iupdt nombre_instancia
```

El parámetro *nombre_instancia* es el nombre de la instancia.

En AIX, este programa de utilidad está situado en el directorio siguiente: /usr/lpp/db2_07_01/instance. Si necesita ayuda, escriba `db2iupdt -h` en la línea de mandatos para abrir el menú de ayuda.

Nota: Actualice el entorno de instancia con Spatial Extender antes de verificar la instalación.

Verificación de la instalación

Después de instalar DB2 Spatial Extender, debe crear una base de datos y ejecutar el programa de verificación de la instalación para comprobar que DB2 Spatial Extender está instalado y configurado correctamente.

Nota: Para las instalaciones en AIX, compruebe que ha definido el entorno de instancia de DB2 Spatial Extender antes de ejecutar el programa de verificación de la instalación.

Puede verificar la instalación utilizando el programa de ejemplo de DB2 Spatial Extender (runGseDemo). Los parámetros de configuración de bases de datos se pueden cambiar desde la línea de mandatos utilizando herramientas de DB2, o mediante la interfaz de usuario proporcionada por el Centro de Control de DB2. Las instrucciones siguientes son válidas para AIX, Windows NT y Windows 2000.

Para verificar la instalación:

1. Inicie la sesión como propietario de la instancia (sólo para AIX).
2. En la configuración del gestor de bases de datos, aumente el tamaño de ocupación de la memoria para las UDF, con un valor mínimo de 2048. Por ejemplo, escriba `db2 update dbm cfg using UDF_MEM_SZ 2048`. Si 2048 no es apropiado, aumente el parámetro UDF_MEM_SZ en incrementos de 256.

   Nota: Las necesidades de memoria para el tamaño de ocupación de la memoria para las UDF aumentan a medida que es mayor el número de las UDF referenciadas en una aplicación. Esto es especialmente cierto cuando las UDF espaciales con tipos de datos espaciales se utilizan como parámetros de entrada y/o salida.
3. Cree una base de datos. Por ejemplo, escriba `db2 create database mydb`, donde *mydb* es el nombre de la base datos.

4. Aumente el tamaño del archivo de anotaciones de DB2 correspondiente a la base de datos.

   Para aumentar el tamaño del archivo de anotaciones:

   a. Conéctese a la base de datos que ha creado. Por ejemplo, escriba `db2 connect to` *`mydb`*, donde *mydb* es el nombre de la base datos.
   b. Aumente el tamaño del archivo de anotaciones. Por ejemplo, escriba `db2 update db logfilesize using LOGFILE 1000`.
   c. Desconéctese de la base de datos. Por ejemplo, escriba `db2 connect reset`.

   Nota: Debe aumentar el tamaño del archivo de anotaciones de DB2 cada vez que habilite una base de datos para operaciones espaciales.

5. Localice el programa de verificación de la instalación. Por ejemplo, escriba `runGseDemo`.

   Para AIX, escriba `cd $HOME/sqllib/samples/spatial`, donde *$HOME* es el directorio inicial del propietario de la instancia.

   Para Windows NT y Windows 2000, escriba `cd c:\sqllib\samples\spatial`, donde *c:\sqllib* es el directorio donde instaló DB2 Spatial Extender.

6. Ejecute el programa de verificación de la instalación. Por ejemplo, escriba:

   `runGseDemo` *`mydb IDusuario contraseña`*

   El parámetro *mydb* es el nombre de la base de datos.

Consejos para la resolución de problemas en el programa de ejemplo

El programa de ejemplo de DB2 está diseñado para poner de manifiesto problemas de la instalación. Durante la verificación de la instalación, el usuario puede recibir mensajes de error que le pueden ayudar a diagnosticar determinados problemas del sistema. La mayoría de los mensajes de error están causados por un pequeño número de errores habituales del usuario. Para evitar estos errores, siga los pasos siguientes cada vez que ejecute el programa de verificación de la instalación:

- Compruebe que ha instalado el cliente y el servidor de DB2 Spatial Extender en los entornos apropiados. En el caso de una configuración autónoma, compruebe que están instalados el cliente y el servidor.
- Utilice una nueva base de datos que no tenga ninguna operación espacial asociada a ella.
- Aumente la configuración del gestor de bases de datos para el tamaño de ocupación de la memoria para las UDF.
- Aumente el tamaño del archivo de anotaciones.

Administration Client

Si no seleccionó la opción **DB2 Administration Client** (para instalaciones en Windows NT) ni la opción **Spatial Extender Client** (para instalaciones en AIX) cuando instaló DB2 Spatial Extender, recibirá el siguiente mensaje de error: ″El nombre especificado no se reconoce como mandato interno ni externo, programa ejecutable ni archivo de proceso por lotes.″

Este error se produce debido a que el sistema no puede acceder al programa de ejemplo. El programa de ejemplo se incluye junto con los productos DB2 Administration Client y Spatial Extender Client. Si DB2 Administration Client o Spatial Extender Client no está instalado en el sistema, entonces el programa de ejemplo tampoco está disponible en el sistema.

Para corregir este problema:

1. Vuelva a instalar DB2 Spatial Extender. Seleccione **DB2 Administration Client** en el programa de configuración de la instalación en Windows NT y Windows 2000, o **Spatial Extender Client** en el programa de instalación de DB2 en AIX.
2. Vuelva a ejecutar el programa de ejemplo repitiendo los pasos de la sección “Verificación de la instalación” en la página 26.

La base de datos ya está habilitada para operaciones espaciales

Recibirá el siguiente mensaje de error si la base de datos para la cual está ejecutando el programa de ejemplo ya está habilitada para operaciones espaciales:

```
Habilitándose la base de datos logtst...
Salida de ENABLE_DB:
Código de retorno = -14
Texto del mensaje de terminación =
GSE0014E  La base de datos ya se ha habilitado para operaciones espaciales.
```

Para corregir este problema, elimine la base de datos y repita los pasos de la sección “Verificación de la instalación” en la página 26.

Nota: Compruebe que la base de datos para la que está verificando la instalación es nueva y no tiene operaciones espaciales asociadas; si las tiene, el programa de ejemplo no se ejecutará correctamente.

Configuración del gestor de bases de datos

Recibirá el siguiente mensaje de error si no aumenta el tamaño de ocupación de la memoria para las UDF en la configuración del gestor de bases de datos:

```
Se ha producido un error de SQL inesperado ("SQL0973N  No hay suficiente
almacenamiento dinámico disponible en "UDF_MEM" para procesar la sentencia").
SQLSTATE=57011
```

Para obtener instrucciones sobre cómo aumentar el tamaño en la configuración del gestor de bases de datos, vea el Paso 2 en la página 26 de la sección "Verificación de la instalación" en la página 26.

Tamaño del archivo de anotaciones

Recibirá el siguiente mensaje de error si no aumenta el tamaño del archivo de anotaciones:

```
Habilitándose la base de datos logtst...
Salida de ENABLE_DB:
Código de retorno = -8
Texto del mensaje de terminación =
GSE0008E  Se ha producido un error inesperado de SQL ("SQL3306N  Se ha
producido el error "-964" de SQL al insertar una fila en  ").
```

Para obtener instrucciones sobre cómo aumentar el tamaño del archivo de anotaciones, vea el Paso 4 en la página 27 en la sección "Verificación de la instalación" en la página 26.

Consideraciones posteriores a la instalación

Después de instalar Spatial Extender, puede realizar lo siguiente:

- Bajar ArcExplorer para Java, Version 3.0
- Utilizar los CD-ROM correspondientes a los datos de referencia y mapas del geocodificador de DB2 Spatial Extender.

Cómo bajar ArcExplorer

ESRI (Environmental Systems Research Institute) proporciona un explorador que puede mostrar los resultados visuales de consultas para datos de DB2 Spatial Extender. Este explorador es ArcExplorer para Java, Versión 3.0. Puede bajar una copia de ArcExplorer para Java, Versión 3.0, desde el sitio Web de ESRI, ubicado en http://www.esri.com. ArcExplorer necesita JRE (Java® Runtime Environment), Versión 1.2.2., Standard Edition o Enterprise Edition.

Para obtener más información sobre cómo instalar y utilizar ArcExplorer para Java, Versión 3.0, consulte el manual *Using ArcExplorer*, que también está disponible en el sitio Web de ESRI.

Importante: Junto con el JDK (Java Development Kit) de IBM, Versión 1.1.8, se proporciona DB2 Universal Database Versión 7.1. Cuando instale JRE 1.2.2 para ArcExplorer, colóquelo en un directorio distinto del de DB2. Recuerde definir la variable de entorno CLASSPATH.

Utilización de los CD-ROM correspondientes a los datos de referencia y mapas del geocodificador de DB2 Spatial Extender

Junto con DB2 Spatial Extender se proporcionan cinco discos CD-ROM de datos y mapas y un CD-ROM de datos para el geocodificador.

Datos y mapas de DB2 Spatial Extender

Los datos y mapas de Spatial Extender están contenidos en cinco discos CD-ROM, cuyas etiquetas llevan la inscripción ″DB2 Spatial Extender Data and Maps 1 – 5″. La Tabla 2 proporciona un resumen de los datos contenidos en cada CD-ROM.

Tabla 2. Información de los discos CD-ROM de datos y mapas

CD-ROM de datos y mapas	Resumen de los tipos de datos de mapa
CD-ROM 1	Canadá, Europa, Méjico, Estados Unidos y mapamundi
CD-ROM 2	Estados Unidos (detallado)
CD-ROM 3	Estados Unidos (región occidental)
CD-ROM 4	Estados Unidos (región oriental)
CD-ROM 5	Estados Unidos (región meridional) e imágenes de ejemplo

Para obtener una descripción detallada de los datos proporcionados por ESRI, consulte el archivo de ayuda de ESRI, esridata.hlp, situado en el CD-ROM de datos y mapas de DB2 Spatial Extender.

- Para Windows NT y Windows 2000, vea el archivo de ayuda situado en *x:* esridata.hlp, donde *x:* es la unidad de CD-ROM.
- Para AIX, visualice o imprima el archivo de ayuda del CD-ROM, situado en */cdrom/esridata.hlp*, donde */cdrom* es el punto de montaje.

Datos de referencia del geocodificador de DB2 Spatial Extender

Los datos de referencia del geocodificador de DB2 Spatial Extender, contenidos en CD-ROM, se han creado específicamente para ser utilizados con el geocodificador por omisión de DB2 Spatial Extender. Estos datos comprenden una red básica de calles de Estados Unidos que el geocodificador por omisión utiliza para determinar la latitud y longitud geográficas de las direcciones contenidas en una base de datos habilitada para operaciones espaciales. Este mapa básico de direcciones son los llamados ″datos de referencia″. El geocodificador por omisión utiliza como entrada direcciones (no espaciales) contenidas en la base de datos del usuario, las compara y asocia con los datos de referencia y las convierte en coordenadas que pueden ser almacenadas por DB2 Spatial Extender. Este proceso se denomina *geocodificación*.

Para obtener más información sobre la geocodificación, consulte la sección “Utilización de geocodificadores” en la página 53.

Acceso a los datos de referencia del geocodificador: Puede acceder directamente a los datos del geocodificador contenidos en el CD-ROM, o puede copiar los datos en la unidad de disco duro. Para copiar los archivos de

datos del geocodificador desde el CD-ROM al entorno servidor de DB2 Spatial Extender, siga los pasos descritos en esta sección.

Para AIX:

1. Monte el CD-ROM. Para saber cómo montar un CD-ROM, consulte la sección "Montaje del CD-ROM" en la página 22.
2. Inicie una sesión como usuario con autorización root en la máquina servidor de destino.
3. Escriba lo siguiente:

   ```
   cp /cdrom/db2/* /usr/lpp/db2_07_01/gse/refdata/
   ```

4. Finalice la sesión.

Para Windows NT y Windows 2000, puede utilizar la ventana de mandatos o el Explorador de Windows.

Para acceder a los datos del geocodificador a través de la ventana de mandatos:

1. Pulse **Inicio** –>**Programa** –> **IBM DB2** –> **Ventana de mandatos**.
2. Escriba lo siguiente:

   ```
   copy d:\db2\* %db2path%\gse\refdata
   ```

   Sustituya *d:* por la letra correspondiente a su unidad de CD-ROM.

Para acceder a los datos del geocodificador a través del Explorador de Windows:

Copie todos los archivos de `d:\db2` a `c:\sqllib\gse\refdata` , donde *d:* es la unidad de CD-ROM y *c:\sqllib* es el directorio donde está instalado DB2.

Suministro del archivo EDGELocator.loc al geocodificador por omisión: Los datos de referencia proporcionados en el CD-ROM incluyen el archivo EDGELocator.loc. El geocodificador por omisión utiliza el archivo EDGELocator.loc para localizar datos de referencia determinados. Por ejemplo, si está geocodificando direcciones de California, Kentucky y Oregón, el geocodificador por omisión utiliza el archivo localizador para determinar las ubicaciones referenciables contenidas en el CD-ROM.

Si habilita la geocodificación incremental (también llamada geocodificación automática) con el geocodificador por omisión, o si utiliza éste en la modalidad de proceso por lotes, debe utilizar el parámetro de entrada **vendorSpecific**. Si especifica el parámetro de entrada **vendorSpecific**, la vía de acceso y el nombre del archivo localizador se deben pasar al geocodificador.

Por ejemplo, el siguiente mandato **gseadm** sirve para invocar la geocodificación en la modalidad de proceso por lotes con el geocodificador por omisión:

```
gseadm run_gc database_name -layerSchema inst1 -layerTable myTable
  -layerColumn column1 -gcId 1
  -vendorSpecific c:\sqlib\gse\refdata\EDGELocator.loc
```

Para obtener más información sobre la utilización del geocodificador y el parámetro **vendorSpecific**, vea el "Capítulo 4. Cómo definir columnas espaciales, registrarlas como capas y habilitar un geocodificador para que las mantenga" en la página 45 y el "Capítulo 5. Cómo llenar columnas espaciales" en la página 53.

Invocación de Spatial Extender

Una vez instalado Spatial Extender, puede utilizar el Centro de Control de DB2 para configurar el entorno GIS y comenzar a trabajar con información espacial.

Para invocar Spatial Extender desde el Centro de Control de DB2:

1. En la ventana del Centro de Control, seleccione el servidor en el que desea utilizar Spatial Extender.
2. Pulse sobre la carpeta **Bases de datos**. Las bases de datos se mostrarán en el panel de contenido.
3. Pulse con el botón derecho del ratón sobre la base de datos con la que desea trabajar y, en el menú emergente, seleccione la operación espacial que desea realizar.

Capítulo 3. Preparación de recursos

Después de instalar Spatial Extender, ya está listo para suministrar a su base de datos los recursos que necesita para crear columnas espaciales y manipular datos espaciales. Este capítulo resume estos recursos y describe dos de las tareas mediante las cuales los recursos pasan a estar disponibles: habilitar la base de datos para operaciones espaciales y crear sistemas de referencias espaciales.

Inventario de recursos

Los recursos que necesita para crear columnas espaciales y para manipular datos espaciales incluyen:

- *Datos de referencia*: direcciones que Spatial Extender comprueba para verificar las direcciones que desea geocodificar.
- Recursos que habilitan una base de datos para operaciones espaciales: procedimientos almacenados, funciones espaciales y otros.
- Geocodificadores, distintos del geocodificador por omisión, que suministran usuarios y proveedores.
- Sistemas de referencias espaciales.

Esta sección trata sobre los datos de referencia y los recursos que habilitan una base de datos para operaciones espaciales. Para obtener información sobre geocodificadores distintos del geocodificador por omisión, consulte la sección "Acerca de la geocodificación" en la página 53. Para obtener información sobre sistemas de referencias espaciales, consulte la sección "Acerca de los sistemas de coordenadas y de referencias espaciales" en la página 35.

Datos de referencia

Los *datos de referencia* consisten en las direcciones más recientes de Estados Unidos que ha recopilado la Oficina de censo de Estados Unidos. Para que el geocodificador por omisión pueda convertir una dirección de la base de datos en coordenadas, debe comparar la dirección o parte de la misma con una dirección de los datos de referencia.

Los datos de referencia están disponibles desde el momento que instala Spatial Extender. Para ver la cantidad de espacio en disco que necesitan estos datos, consulte la sección "Requisitos de espacio de disco" en la página 19. Para comprobar en AIX que los datos se han cargado correctamente, búsquelos en el directorio $DB2INSTANCE/sqllib/gse/refdata/. Para comprobar en Windows NT que los datos se han cargado correctamente, búsquelos en el directorio %DB2PATH%\gse\refdata\.

Recursos que habilitan una base de datos para operaciones espaciales

La primera tarea a llevar a cabo después de instalar Spatial Extender consiste en habilitar la base de datos para operaciones espaciales. Esto implica iniciar una acción que haga que Spatial Extender cargue la base de datos con los siguientes recursos:

- Procedimientos almacenados. Cuando solicita una acción del Centro de Control, Spatial Extender invoca uno de estos procedimientos almacenados para ejecutar la acción.
- Tipos de datos espaciales. Debe asignar un tipo de datos espaciales a cada columna de vista o tabla en la que se van a almacenar datos espaciales. Para obtener más información, consulte la sección "Acerca de los tipos de datos espaciales" en la página 45.
- Vistas y tablas de catálogo de Spatial Extender. Ciertas operaciones dependen del catálogo de Spatial Extender. Por ejemplo, para que se pueda llenar una columna con un tipo de datos espaciales, se debe registrar en el catálogo como una capa. Para obtener información sobre capas, consulte la sección "Desarrollo y aplicación de un proyecto GIS" en la página 10.
- Un tipo de índice espacial. Le permite definir índices para capas.
- Funciones espaciales. Sirven para trabajar con datos espaciales de diversas formas; por ejemplo, para determinar las relaciones entre elementos geográficos y para generar más datos espaciales. Una de estas funciones es el geocodificador por omisión. El geocodificador convierte direcciones de Estados Unidos en coordenadas y luego inserta estas coordenadas en columnas espaciales. Para obtener más información sobre funciones espaciales, consulte el "Capítulo 13. Geometrías y funciones espaciales asociadas" en la página 159 y el "Capítulo 14. Funciones espaciales para consultas SQL" en la página 199. Para obtener más información sobre el geocodificador por omisión, consulte la sección "Acerca de la geocodificación" en la página 53.
- Un esquema, denominado DB2GSE, que contiene los objetos que se acaban de listar.

Para obtener instrucciones sobre cómo utilizar el Centro de Control para iniciar la carga de estos recursos, consulte la sección "Habilitación de una base de datos para operaciones espaciales" en la página 35. Para obtener instrucciones sobre cómo utilizar una rutina de un programa de aplicación para realizar la misma tarea, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Habilitación de una base de datos para operaciones espaciales

Para saber qué autorización se necesita para habilitar una base de datos para operaciones espaciales, consulte la sección "Autorización" en la página 95.

Para habilitar una base de datos para operaciones espaciales desde el Centro de Control:

1. Desde la ventana Centro de Control, amplíe el árbol de objetos hasta encontrar la carpeta **Bases de datos** bajo el servidor en el que desea ejecutar Spatial Extender.
2. Pulse la carpeta **Bases de datos**. Se muestran las bases de datos en el panel de contenido en la parte derecha de la ventana.
3. Pulse con el botón derecho del ratón la base de datos que desee y pulse **Spatial Extender —> Habilitar** en el menú emergente. Spatial Extender suministra a la base de datos los recursos que le permiten crear columnas y datos espaciales y trabajar con los mismos.

Recuerde: Para poder habilitar una base de datos para operaciones espaciales, Spatial Extender debe estar instalado en el servidor en el que reside la base de datos.

Creación de un sistema de referencias espaciales

Esta sección describe la relación entre sistemas de referencias espaciales y sistemas de coordenadas y explica cómo crear un sistema de referencias espaciales desde el Centro de Control.

Acerca de los sistemas de coordenadas y de referencias espaciales

Esta sección continúa la explicación sobre sistemas de coordenadas iniciada en el tema "La naturaleza de los datos espaciales" en la página 6. Luego explica con más detalle los sistemas de referencias espaciales definidos en el tema "Desarrollo y aplicación de un proyecto GIS" en la página 10. También contiene instrucciones para determinar qué valores asignar a los parámetros del sistema de referencias espaciales.

Sistemas de coordenadas, coordenadas y medidas

Puede considerar que un sistema de coordenadas es una cuadrícula imaginaria que abarca una determinada zona geográfica; por ejemplo, una cuadrícula que abarca la superficie terrestre, un país o una región de un estado. Cada elemento geográfico de la zona está situado en la intersección de una línea de cuadrícula este-oeste y una línea de cuadrícula norte-sur. Un valor, denominado *coordenada X*, indica dónde está situado el elemento en la línea de cuadrícula este-oeste. Otro valor, denominado *coordenada Y*, indica dónde está situado el elemento en la línea de cuadrícula norte-sur. Ambos valores denotan la posición con respecto al centro u *origen* de la cuadrícula.

Las coordenadas X e Y del origen son ambas cero. A partir del origen hacia el este, las coordenadas X son positivas; a partir del origen hacia el oeste, son negativas. Similarmente, a partir del origen hacia el norte, las coordenadas Y son positivas; a partir del origen hacia el sur, son negativas. Considere, por ejemplo, un sistema de coordenadas A, que incluye una cuadrícula que cubre una gran zona metropolitana. Una coordenada X de 7 indicaría una posición que está siete unidades al este del origen de esta cuadrícula. Una coordenada X de -9,5 indicaría una posición que está nueve unidades y media al oeste del origen.

Cada dato elemental de una columna espacial incluye (1) una coordenada X y una coordenada Y que definen la ubicación de un elemento geográfico o (2) varias coordenadas X e Y que definen las ubicaciones de partes de un elemento geográfico o que definen la zona cubierta por un elemento. También se pueden incluir otros dos tipos de valores: una *coordenada Z* y una *medida*. A diferencia de las coordenadas X e Y, las coordenadas Z y las medidas no se utilizan en Spatial Extender para definir ubicaciones o zonas. Simplemente proporcionan información necesaria para una aplicación GIS. Una coordenada Z suele indicar la altura o profundidad de un elemento geográfico. Las coordenadas Z que están por encima del origen son positivas; las que están por debajo son negativas. Una medida es numérica; puede contener cualquier tipo de información. Por ejemplo, supongamos que está representando pozos de petróleo en su GIS. Si necesita que las aplicaciones procesen valores que indican ID de puntos identificados correspondientes a datos sísmicos, puede almacenar estos valores como medidas.

Sistemas de referencias espaciales, desplazamientos y factores de escala

Tal como se indica en el tema “Sistemas de coordenadas, coordenadas y medidas” en la página 35, las coordenadas pueden ser negativas y se pueden expresar en decimales. Esto también se aplica a las medidas. Sin embargo, para reducir la carga general del almacenamiento, Spatial Extender almacena cada coordenada y medida como un entero no negativo (es decir, como un entero positivo o cero). Por lo tanto, las coordenadas y medidas reales negativas y decimales se deben convertir en enteros no negativos para que Spatial Extender las pueda guardar. Además, debe indicar a Spatial Extender cómo realizar la conversión. Para ello debe definir ciertos parámetros. Los valores de los parámetros que se utilizarán para convertir coordenadas y medidas dentro de un área geográfica se denominan de forma colectiva *sistema de referencias espaciales*.

Puede crear un sistema de referencias espaciales siguiendo estos pasos:

- Primero, determine las coordenadas y medidas negativas más bajas de los elementos geográficos que está representando. (Cuanto más lejos de cero

está un valor negativo, menor es. Una coordenada X de –10 es menor que una coordenada X de –5; una medida de –100 es menor que una medida de –50.)

- Especifique *factores de desplazamiento* (o *desplazamientos*, para abreviar): valores que, cuando se restan de medidas y coordenadas negativas, dan como resultado números no negativos.
- Especifique *factores de escala*: valores que, cuando se multiplican por medidas y coordenadas decimales, dan lugar a enteros cuya precisión es como mínimo igual a la de las coordenadas o medidas. Por ejemplo, supongamos que tenemos una coordenada con una precisión de cuatro: 92,77. Podría multiplicarla por un factor de escala de 100 para obtener un entero con una precisión de cuatro: 9277. Observe que cuando se crea un sistema de referencias espaciales, los factores de desplazamiento se aplican antes que los factores de escala.

Determinación de las coordenadas y medidas negativas más bajas

DB2 Spatial Extender puede guardar coordenadas y medidas si son valores positivos enteros, pero no si son números negativos o decimales. Por tanto, es necesario convertir las coordenadas y medidas negativas en valores positivos, y las coordenadas y medidas decimales en valores enteros. Para realizar esta conversión, debe definir un conjunto de parámetros que, cuando se aplican a medidas de coordenadas negativas o decimales, dan lugar a enteros positivos. Este conjunto de parámetros se denomina *sistema de referencias espaciales.* Los parámetros utilizados para convertir valores negativos se denominan *factores de desplazamiento*; los utilizados para convertir valores decimales se denominan *factores de escala.*

Cuando invoca una función espacial que utiliza como entrada una coordenada o medida decimal, y el identificador de un sistema de referencias espaciales, la función multiplica la coordenada o medida decimal por un factor de escala existente en el sistema. El resultado es un entero que DB2 Spatial Extender almacena. El factor de escala debe ser lo suficiente grande para asegurar que la precisión del entero resultante sea la misma que la precisión de la coordenada decimal.

Por ejemplo, suponga que la entrada de la función ST_Point consta de los valores 10.01 como coordenada X, 20.03 como coordenada Y, y el identificador del sistema de referencias espaciales. Cuando se invoca ST_Point, la función multiplica el valor 10.01 y el valor 20.03 por el factor de escala que el sistema de referencias espaciales proporciona para las coordenadas X e Y. Si este factor de escala es 10, los valores resultantes respectivos que DB2 Spatial Extender almacenará serán 100 y 200. Debido a que la precisión de estos enteros (3) es menor que la precisión de las coordenadas (4), DB2 Spatial Extender no podrá revertir estos enteros a las coordenadas originales ni obtener a partir de ellos valores que sean coherentes con el sistema de coordenadas al que pertenecen estas coordenadas. Pero si el factor de escala es 100, los enteros resultantes

que DB2 Spatial Extender almacenará serán 1001 y 2003, que son valores que se pueden revertir a las coordenadas originales o de los que se pueden obtener coordenadas compatibles.

Para poder definir parámetros para un sistema de referencias espaciales, tiene que determinar las coordenadas X, Y, Z y la medida más bajas del área geográfica que contienen los elementos sobre los que desea información. Puede buscar estos valores contestando las siguientes preguntas:

- Entre los elementos que está representando, ¿hay alguno que esté al oeste del origen del sistema de coordenadas utilizado? Si es así, ¿qué coordenada X indica la ubicación de la esquina oeste del elemento situado más hacia el oeste? (La respuesta será la coordenada X negativa más baja que está utilizando.) Por ejemplo, si está representando pozos de petróleo, y algunos quedan al oeste del origen, ¿qué coordenada X indica la ubicación del pozo de petróleo situado más hacia el oeste?
- ¿Hay algún elemento al sur del origen? Si es así, ¿qué coordenada Y indica la ubicación de la esquina sur del elemento situado más hacia el sur? (La respuesta será la coordenada Y negativa más baja que está utilizando.) Por ejemplo, si está representando pozos de petróleo, y algunos están al sur del origen, ¿qué coordenada Y indica la ubicación del pozo de petróleo situado más hacia el sur?
- Si va a utilizar coordenadas Z para definir profundidades, ¿qué elemento geográfico es el más profundo y qué coordenada Z representa el punto más bajo de ese elemento? (La respuesta será la coordenada Z negativa más baja que está utilizando.)
- Si va a incluir medidas en sus datos espaciales ¿habrá alguna negativa? Si es así, ¿cuál es la medida negativa más baja?

Una vez determinadas las coordenadas y medidas negativas más bajas, sume a cada una de ellas una cantidad igual a entre el cinco y el diez por ciento de su valor. Por ejemplo, si la coordenada X negativa más baja es –100, podría sumarle –5. En este manual se denomina al número resultante un *valor aumentado*.

Nota: El identificador del sistema de referencias espaciales por omisión de DB2 Spatial Extender es 0 (cero). DB2 Spatial Extender proporciona un sistema de referencias espaciales para su utilización con el geocodificador por omisión. El identificador de este sistema de referencias es 1.

Especificación de factores de desplazamiento

A continuación, especifique qué factores de desplazamiento debe utilizar Spatial Extender para convertir coordenadas y medidas negativas en coordenadas y medidas no negativas:

- Después de decidir qué valor aumentado de X desea, especifique un desplazamiento que, cuando se reste de ese valor, dé como resultado cero. Spatial Extender restará este número de todas las coordenadas X negativas para llegar a un valor positivo. Spatial Extender también restará este número de todas las demás coordenadas X.

  Por ejemplo, si el valor aumentado de X es –105, tiene que restarle –105 para conseguir un valor igual a 0. Luego Spatial Extender restará –105 de todas las coordenadas X asociadas a los elementos representados. Puesto que ninguna de estas coordenadas es mayor que –100, todos los valores resultantes de la resta serán positivos.
- Paralelamente, especifique desplazamientos que den como resultado 0 cuando se resten del valor aumentado de Y, del valor aumentado de Z y de la medida aumentada.

El desplazamiento restado de coordenadas X se denomina *X falsa*. Los desplazamientos restados de coordenadas Y, de coordenadas Z y de medidas se denominan *Y falsa*, *Z falsa* y *M falsa* respectivamente. Para ver instrucciones sobre cómo especificar estos parámetros desde el Centro de Control, consulte la sección "Creación de un sistema de referencias espaciales desde el Centro de Control" en la página 40.

Especificación de factores de escala

A continuación, especifique qué factores de escala debe utilizar Spatial Extender para convertir medidas y coordenadas decimales en enteros:

- Especifique un factor de escala que, cuando se multiplique por una coordenada X decimal o por una coordenada Y decimal, dé como resultado un entero de 32 bits. Se recomienda que este factor de escala sea un factor de 10: 10 a la primera potencia (10), 10 a la segunda potencia (100), 10 a la tercera potencia (1000), o, si es necesario, un factor mayor. Para decidir qué factor de 10 debe ser el factor de escala:
  1. Determine qué coordenadas X e Y son números decimales o tienen bastante probabilidad de serlo. Por ejemplo, supongamos que determina que tres de las diversas coordenadas X e Y que va a manejar son números decimales: 1,23, 5,1235 y 6,789.
  2. Tome la coordenada decimal que tiene la precisión decimal más larga. Luego determine por qué factor de diez se puede multiplicar esta coordenada para obtener un entero de igual precisión. Para ilustrar: de las tres coordenadas decimales del ejemplo actual, 5,1235 tiene la precisión decimal más larga. Si la multiplicamos por diez a la cuarta potencia (10000), obtendremos un entero, 51235.
  3. Determine qué entero generado por la multiplicación que se acaba de describir es demasiado largo para almacenarlo como un elemento de datos de 32 bits. 51235 no es demasiado largo. Pero supongamos que, además de 1,23, 5,11235 y 6,789, su rango de coordenadas X e Y incluye un cuarto valor decimal, 10006,789876. Puesto que la precisión decimal

de esta coordenada es más larga que las de las otras tres, tendría que multiplicar *esta* coordenada—no 5,1235—por un factor de 10. Para convertirla en un entero, la tendría que multiplicar por 10 a la sexta potencia (1000000). Pero el valor resultante, 10006789876, es demasiado largo para almacenarlo como un elemento de datos de 32 bits. Si Spatial Extender intentara almacenarlo, los resultados serían imprevisibles.

Para evitar este problema, seleccione un factor de 10 que, multiplicado por la coordenada original, dé como resultado un número decimal que Spatial Extender pueda truncar en un entero que pueda almacenar, con la mínima pérdida de precisión. En este caso, podría seleccionar 10 a la cuarta potencia (10000). El resultado de multiplicar 10000 por 10006,789876 es 100067898,76. Spatial Extender truncaría este número a 100067898, reduciendo su precisión por una cantidad virtualmente insignificante.

- Si los elementos geográficos representados tienen coordenadas Z decimales, siga el procedimiento anterior para determinar un factor de escala para estas coordenadas. Si los elementos están asociados a medidas decimales, siga el mismo procedimiento para determinar un factor de escala para estas medidas.

El factor de escala para las coordenadas X e Y se denomina *unidad XY*. Los factores de escala para coordenadas Z y medidas se denominan *unidades Z* y *unidades M*, respectivamente. Para ver instrucciones sobre cómo especificar estos parámetros desde el Centro de Control, consulte la sección "Creación de un sistema de referencias espaciales desde el Centro de Control".

Creación de un sistema de referencias espaciales desde el Centro de Control

Esta sección contiene una visión general de los pasos a seguir para crear un sistema de referencias espaciales desde el Centro de Control. La visión general va seguida de detalles sobre cómo realizar cada paso.

No es necesaria ninguna autorización para ejecutar estos pasos.

Visión general para crear un sistema de referencias espaciales desde el Centro de Control:

1. Abra la ventana Crear Sistema de Referencias Espaciales.
2. Indique qué sistema de coordenadas desea utilizar.
3. Especifique los identificadores del sistema de referencias espaciales que desea crear.
4. Determine qué rangos de coordenadas y medidas se aplican a los elementos geográficos sobre los que desea información.

5. Especifique valores que se puedan utilizar para convertir coordenadas y medidas negativas o decimales en datos elementales que Spatial Extender pueda almacenar.
6. Indique a Spatial Extender que cree el sistema de referencias espaciales que desee.

Pasos detallados para crear un sistema de referencias espaciales desde el Centro de Control:

1. Abra la ventana Crear Sistema de Referencias Espaciales.
   a. Desde la ventana del Centro de Control, amplíe el árbol de objetos hasta encontrar la carpeta **Bases de datos** bajo el servidor en el que desea ejecutar Spatial Extender.
   b. Pulse la carpeta **Bases de datos**. Se muestran las bases de datos en el panel de contenido en la parte derecha de la ventana.
   c. Pulse con el botón derecho del ratón la base de datos que ha habilitado para datos espaciales y pulse **Spatial Extender —> Referencias espaciales** en el menú emergente. Se abrirá la ventana Referencias espaciales.
   d. En la ventana Referencias espaciales, pulse **Crear**. Se abrirá la ventana Crear Sistema de Referencias Espaciales.
2. En la ventana Crear Sistema de Referencias Espaciales, utilice el campo **Sistema de coordenadas** para indicar qué sistema de coordenadas desea utilizar.
3. Especifique los identificadores del sistema de referencias espaciales que desea crear.
   - En el campo **Nombre**, escriba un nombre de entre 1 y 64 caracteres para el sistema.

     Restricción: No especifique el nombre de otro sistema de referencias espaciales. En la base de datos no puede haber dos sistemas de referencias espaciales con el mismo nombre.
   - En el campo ID, escriba un identificador numérico. Debe ser un entero.

     Restricción: No especifique el ID de otro sistema de referencias espaciales. En la base de datos no puede haber dos sistemas de referencias espaciales con el mismo ID.
4. En un medio externo al Centro de Control—como por ejemplo un papel o una pizarra—determine las coordenadas y medidas negativas más bajas que se aplican a los elementos geográficos que está representando. Para obtener instrucciones sobre cómo hacerlo, consulte la sección "Determinación de las coordenadas y medidas negativas más bajas" en la página 37.

5. En la ventana Crear Sistema de Referencias Espaciales, especifique valores para convertir coordenadas y medidas negativas o decimales en elementos de datos que Spatial Extender pueda utilizar (es decir, valores enteros no negativos de 32 bits).
   a. Especifique valores para convertir coordenadas X negativas o decimales en enteros no negativos:
      - En la columna **Desplazamiento**, en el campo más cercano a la **X**, especifique una X falsa:
        – Si alguno de los valores dentro del rango de coordenadas X identificado en el paso 4 en la página 41 es negativo, escriba una X falsa que, cuando se reste de la coordenada negativa más baja, dé como resultado un número positivo. Para obtener instrucciones, consulte la sección "Especificación de factores de desplazamiento" en la página 38.
        – Si todas las coordenadas X son no negativas, escriba una X falsa de 0.
      - En la columna **Factor de escala**, especifique una unidad XY en el campo que hay más a la derecha de la **X**. Esta unidad XY debe ser una que, multiplicada por cualquier coordenada X decimal o coordenada Y decimal, dé como resultado un número entero que se pueda almacenar como un elemento de datos de 32 bits, con la mínima pérdida de precisión. Para obtener instrucciones, consulte la sección "Especificación de factores de escala" en la página 39.

        Después de especificar la unidad XY en el campo que hay más a la derecha de la **X**, aparecerá también en el campo que hay más a la derecha de la **Y**.
   b. Especifique una Y falsa que permita a Spatial Extender convertir coordenadas Y negativas en valores positivos. Esto se especifica en la columna **Desplazamiento**, en el campo más cercano a la **Y**:
      - Si alguno de los valores dentro del rango de coordenadas Y identificado en el paso 4 en la página 41 es negativo, escriba una Y falsa que, cuando se reste de la coordenada negativa más baja, dé como resultado un número positivo. Para obtener instrucciones, consulte la sección "Especificación de factores de desplazamiento" en la página 38.
      - Si todas las coordenadas Y son positivas, escriba una Y falsa de 0.
   c. Si va a incluir coordenadas Z en sus datos espaciales, especifique valores para convertir coordenadas Z decimales o negativas en enteros no negativos:
      - En la columna **Desplazamiento**, en el campo más cercano a la **Z**, escriba una Z falsa:
        – Si alguno de los valores dentro del rango de coordenadas Z identificado en el paso 4 en la página 41 es negativo, escriba una

Z falsa que, cuando se reste de la coordenada negativa más baja, dé como resultado un número positivo. Para obtener instrucciones, consulte la sección "Especificación de factores de desplazamiento" en la página 38.

– Si todas las coordenadas Z son no negativas, escriba una Z falsa de 0.

• En la columna **Factor de escala**, especifique una unidad Z en el campo que hay más a la derecha de la **Z**. Esta unidad Z debe ser una que, multiplicada por cualquier coordenada Z decimal, dé como resultado un número entero que se pueda almacenar como un elemento de datos de 32 bits, con la mínima pérdida de precisión. Para obtener instrucciones, consulte la sección "Especificación de factores de escala" en la página 39.

d. Si va a incluir medidas en sus datos espaciales, especifique valores para convertir medidas decimales o negativas en enteros positivos:

• En la columna **Desplazamiento**, en el campo más cercano a la etiqueta **Lineal**, escriba una M falsa:

– Si alguno de los valores dentro del rango de medidas identificado en el paso 4 en la página 41 es negativo, escriba una M falsa que, restada de la medida negativa menor, dé como resultado un número positivo. Para obtener instrucciones, consulte la sección "Especificación de factores de desplazamiento" en la página 38.

– Si todas las medidas son positivas, escriba una M falsa de 0.

• En la columna **Factor de escala**, especifique una unidad M en el campo que hay más a la derecha de la etiqueta **Lineal**. Esta unidad M debe ser una que, multiplicada por cualquier medida decimal, dé como resultado un número entero que se pueda almacenar como un elemento de datos de 32 bits, con la mínima pérdida de precisión. Para obtener instrucciones, consulte la sección "Especificación de factores de escala" en la página 39.

6. Pulse **Bien** para crear el sistema de referencias espaciales que desea.

Capítulo 4. Cómo definir columnas espaciales, registrarlas como capas y habilitar un geocodificador para que las mantenga

Una vez preparados los recursos para su GIS de Spatial Extender, ya está listo para crear objetos que contendrán datos espaciales. Por ejemplo, si necesita nuevas tablas para que contengan datos espaciales, las puede definir asignando tipos de datos espaciales a las columnas en las que desea colocar los datos. Si necesita añadir columnas espaciales a tablas existentes, también puede hacerlo.

Cuando suministra a una tabla nueva o existente una columna espacial, tiene que registrar dicha columna como una capa. Además, si tiene intención de que un geocodificador llene la columna, puede, en el momento de registrar la columna como una capa, habilitar el geocodificador para que la mantenga de forma automática. Esta habilitación se produce del siguiente modo: Spatial Extender define activadores que están codificados para invocar el geocodificador siempre que la columna o columnas de atributo correspondientes a la columna espacial reciben datos nuevos o actualizados. Cuando se le invoca, el geocodificador convierte los datos nuevos o actualizados en datos espaciales y los coloca en la columna espacial.

Después de definir una columna espacial para una tabla puede, si lo desea, crear una columna de vista sobre esta columna de tabla. Debe registrar la columna de vista como una capa después de registrar la columna de la tabla como una capa.

Este capítulo describe la naturaleza y el uso de los tipos de datos que puede asignar a una columna espacial. A continuación, el capítulo explica cómo utilizar el Centro de Control para definir una columna espacial para una tabla, para registrar esta columna como una capa y para habilitar un geocodificador para que la mantenga. Finalmente, el capítulo explica cómo utilizar el Centro de Control para registrar una columna de vista como una capa.

Acerca de los tipos de datos espaciales

Esta sección contiene una introducción a los tipos de datos necesarios para las columnas espaciales y ofrece directrices para seleccionar cuál debe ser el tipo de datos de una columna espacial.

Cuando habilita una base de datos para operaciones espaciales, Spatial Extender suministra a la base de datos una jerarquía de tipos de datos

estructurados. La Figura 6 presenta esta jerarquía. En esta figura, los tipos replicables tienen un fondo blanco; los tipos no replicables tienen un fondo gris.

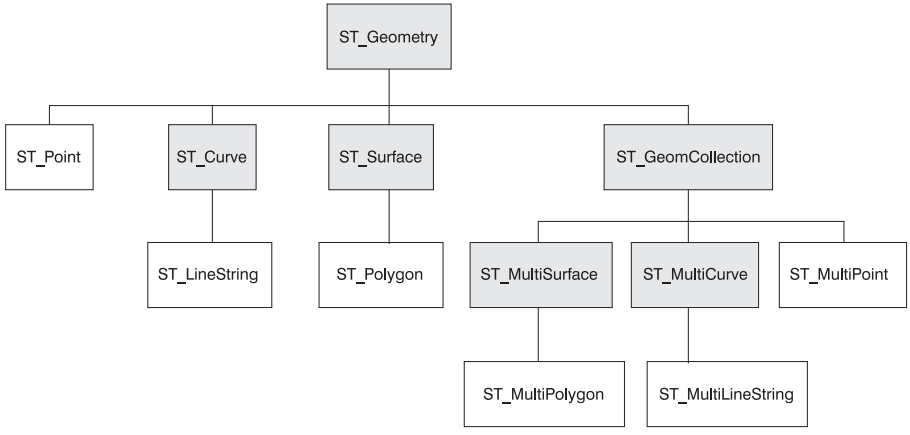


Figura 6. Jerarquía de tipos de datos espaciales. Los tipos de datos que aparecen en los cuadros blancos son replicables. Los tipos de datos que aparecen en los cuadros sombreados no son replicables.

La jerarquía de la Figura 6 incluye:

- Tipos de datos para elementos geográficos que se pueden percibir como una unidad; por ejemplo, edificios individuales y lagos aislados.
- Tipos de datos para elementos geográficos que constan de varias unidades o componentes; por ejemplo, sistemas de autopistas y cadenas de montañas.
- Un tipo de datos para elementos geográficos de todo tipo.

Tipos de datos para elementos geográficos de una sola unidad

Utilice ST_Point, ST_Linestring y ST_Polygon para almacenar coordenadas que definen el espacio ocupado por elementos geográficos que se pueden percibir como formados por una sola unidad:

- Utilice ST_Point cuando desee indicar el punto del espacio ocupado por un elemento geográfico discreto. El elemento geográfico puede ser muy pequeño (un pozo de agua), muy grande (una ciudad) o de tamaño intermedio (un complejo de edificios o un parque). En cada caso, el punto en el espacio se puede situar en la intersección de una línea de coordenadas este-oeste (por ejemplo, un paralelo) y una línea de coordenadas norte-sur (por ejemplo, un meridiano). Un dato elemental de ST_Point incluye valores (una coordenada X y una coordenada Y) que definen una intersección de ese tipo. La coordenada X indica el lugar donde se encuentra la intersección en la línea este-oeste; la coordenada Y indica el lugar donde se encuentra la intersección en la línea norte-sur.

- Utilice ST_Linestring para coordenadas que definen el espacio ocupado por elementos geográficos lineales; por ejemplo, calles, canales y conductos.
- Utilice ST_Polygon cuando desee indicar la extensión de espacio ocupada por un elemento geográfico que consta de varios lados; por ejemplo, un barrio de una ciudad, un bosque o un hábitat salvaje. Un dato elemental de ST_Polygon consta de las coordenadas que definen el perímetro de un elemento geográfico de este tipo.

  En algunos casos, se puede utilizar ST_Polygon y ST_Point para un mismo elemento geográfico. Por ejemplo, supongamos que necesita información espacial referente a varios complejos de apartamentos. Si desea representar el punto del espacio en el que se encuentra cada complejo, utilizaría ST_Point para guardar las coordenadas X e Y que definen ese punto. Por otro lado, si desea representar la zona cubierta por cada complejo, utilizaría ST_Polygon para guardar las coordenadas que definen el perímetro de cada zona.

Tipos de datos para elementos geográficos formados por varias unidades

Utilice ST_MultiPoint, ST_MultiLineString y ST_MultiPolygon para guardar coordenadas que definen espacios ocupados por elementos geográficos formados por varias unidades:

- Utilice ST_MultiPoint si desea representar elementos geográficos formados por unidades discretas y desea indicar el punto en el espacio ocupado por cada componente. Un dato elemental de ST_MultiPoint incluye los pares de coordenadas X e Y que definen la ubicación de cada componente de este tipo de elemento geográfico. Por ejemplo, considere una tabla cuyas filas representan cadenas de islas y cuyas columnas incluyen una columna ST_MultiPoint. Cada dato elemental de esta columna incluye los pares de coordenadas X e Y que definen las ubicaciones de las islas en cada cadena.
- Utilice ST_MultiLineString cuando desee representar elementos geográficos formados por unidades lineales y desee información sobre el espacio ocupado por cada unidad. Un dato elemental de ST_MultiLineString consta de las coordenadas que definen este tipo de espacios. Por ejemplo, considere una tabla cuyas filas representan sistemas de ríos y cuyas columnas incluyen una columna ST_MultiLineString. Cada dato elemental de esta columna incluye los conjuntos de coordenadas que definen los recorridos de los ríos de cada sistema.
- Utilice ST_MultiPolygon cuando desee representar elementos geográficos formados por unidades de varios lados y desee información sobre el espacio ocupado por cada unidad. Por ejemplo, considere una tabla cuyas filas representan condados del medio-oeste americano y cuyas columnas incluyen una columna ST_MultiPolygon. Esta columna contiene información sobre tierras de labranza. En concreto, cada dato elemental de la columna incluye los conjuntos de coordenadas que definen los perímetros de las tierras de labranza de un determinado condado.

Un tipo de datos para todos los elementos geográficos

Puede utilizar ST_Geometry cuando no conozca con seguridad cuál de los demás tipos de datos debe utilizar. Puesto que ST_Geometry es la raíz de la jerarquía a la que pertenecen los demás tipos de datos, una columna ST_Geometry puede contener cualquiera de los valores (o todos ellos) que se pueden almacenar en columnas a las que están asignados los demás tipos de datos.

Atención: Si piensa utilizar el geocodificador por omisión para llenar con datos una columna espacial, la columna debe ser de tipo ST_Point o ST_Geometry.

Cómo definir una columna espacial para una tabla, registrar esta columna como una capa y habilitar un geocodificador para que la mantenga

Esta sección contiene una visión general de los pasos a seguir para definir una columna espacial para una tabla, registrar esta columna como una capa y habilitar un geocodificador para que la mantenga. La visión general va seguida de detalles sobre cómo llevar a cabo cada paso.

Para saber qué autorización necesita para registrar una columna de tabla como una capa, consulte la sección "Autorización" en la página 110. Para saber qué autorización necesita para habilitar un geocodificador para que mantenga esta columna, consulte la sección "Autorización" en la página 91.

Visión general de los pasos a seguir para definir una columna espacial para una tabla, registrar esta columna como una capa y habilitar un geocodificador para que la mantenga:

1. Si la columna espacial debe formar parte de una tabla nueva, cree dicha tabla.
2. Abra la ventana Crear Capa Espacial.
3. Añada una columna espacial a una tabla e indique que desea registrar esta columna como una capa, o bien indique que desea registrar una columna existente como una capa.
4. Indique qué sistema de referencias espaciales se debe utilizar para la capa.
5. Si la capa debe contener datos importados o datos generados a partir de otra columna espacial, indique a Spatial Extender que cree la capa.
6. Si la capa debe contener datos obtenidos a partir de datos de atributo:
   a. Especifique qué columna o columnas contienen estos datos de atributo.
   b. Indique que desea habilitar un geocodificador para que mantenga la capa.
   c. Indique a Spatial Extender que cree la capa.

Detalles paso a paso para definir una columna espacial para una tabla, registrar esta columna como una capa y habilitar un geocodificador para que la mantenga:

1. Si la columna espacial debe formar parte de una tabla nueva, cree dicha tabla:
   - Utilice la interfaz que desee (por ejemplo, el Centro de Control o el procesador de línea de mandatos) para crear la tabla.
   - Si tiene intención de utilizar un geocodificador, incluya entre una y diez columnas sobre las que vaya a funcionar el geocodificador. Un geocodificador no puede tomar más de diez columnas de datos como entrada.
   - Incluya la columna espacial que vaya a registrar como una capa o defina esta columna en el paso 3.

   Si desea utilizar una tabla existente, continúe en el paso siguiente.
2. Abra la ventana Crear Capa Espacial.
   a. Desde la ventana Centro de Control, amplíe el árbol de objetos hasta encontrar la carpeta **Tablas** en la que están las tablas de la base de datos que utiliza para operaciones espaciales.
   b. Pulse la carpeta **Tablas**. Se mostrarán las tablas en el panel de contenido de la parte derecha de la ventana.
   c. Pulse con el botón derecho del ratón la tabla que desee y pulse **Spatial Extender** —> **Capas Espaciales** en el menú emergente. Se abrirá la ventana Capas Espaciales.
   d. En la ventana Capas Espaciales, pulse **Crear**. Se abrirá la ventana Crear Capa Espacial.
3. En la ventana Crear Capa Espacial, añada una columna espacial a una tabla, e indique que desea registrar esta columna como una capa, o bien indique que desea registrar una columna existente como una capa.
   - Si desea añadir una columna espacial a una tabla y definir dicha columna como una capa:
     a. En el campo **Columna de capa**, escriba un nombre para la columna.
     b. En el campo **Tipo de columna**, seleccione o escriba el tipo de datos que desea que tenga la columna. Para ver los tipos de datos permitidos, consulte la sección "Acerca de los tipos de datos espaciales" en la página 45.
   - Si desea definir una columna existente como una capa, selecciónela en el campo **Columna de capa**.

     Restricción: No seleccione una columna que ya se haya definido como una capa.

4. En el campo **Nombre de referencia espacial**, especifique el nombre del sistema de referencias espaciales a utilizar para la capa.
5. Si desea que la capa contenga datos importados o datos generados a partir de otra columna espacial, pulse **Bien** para registrarla.
6. Si desea que la capa contenga datos obtenidos a partir de datos de atributo:
   a. Especifique qué columna o columnas contienen estos datos de atributo:
      1) Seleccione la columna o columnas en el recuadro **Columnas disponibles**. Puede seleccionar un máximo de diez columnas.
      2) Pulse el botón >, el botón >> o ambos para listar la columna o columnas seleccionadas en el recuadro **Columnas seleccionadas**.
   b. Si desea habilitar un geocodificador para que mantenga la capa:
      1) Seleccione el recuadro de selección **Habilitar geocodificador automático**.
      2) En el campo **Nombre**, seleccione el nombre del geocodificador que desea utilizar.
      3) En el campo **Nivel de precisión**, especifique, en términos de porcentaje, el grado en que los registros de entrada deben coincidir con los registros correspondientes de los datos de referencia para que se procesen. Este porcentaje se denomina *precisión*. Por ejemplo, supongamos que el geocodificador lee un registro de entrada que contiene la dirección 557 Bailey, San Jose 94120. Si la precisión es 100 y la coincidencia entre esta dirección y su correspondiente en los datos de referencia no es del 100 por cien, el geocodificador la rechazará. Si la precisión es 75 y la coincidencia entre el registro y su correspondiente en los datos de referencia es de al menos el 75 por ciento, el geocodificador lo procesará.
      4) Si el geocodificador procede de un proveedor, utilice el recuadro **Propiedades** para especificar los parámetros de geocodificación suministrados por el proveedor que desee utilizar.
   c. Pulse **Bien** para registrar la columna seleccionada como una capa y, si lo ha solicitado, para habilitar el geocodificador para que mantenga la columna.

Restricciones

No puede cambiar las columnas que el geocodificador utiliza como entrada. Sin embargo, puede modificar otras propiedades de la geocodificación. Por ejemplo, puede habilitar o inhabilitar la geocodificación incremental.

Nota: La ventana Suprimir Capa no le permite suprimir capas, sino cancelar su registro, es decir, eliminar toda la información sobre ellas que está contenida en el catálogo del sistema de DB2 Spatial Extender. Cuando

cancela el registro de una capa, la columna de tabla o de vista que se utilizó para la capa sigue existiendo.

Cómo registrar una columna de vista como una capa

Para ver qué autorización necesita para registrar una columna de vista como una capa, consulte la sección "Autorización" en la página 110.

Para registrar una columna de vista como una capa:

1. Abra la ventana Crear Capa Espacial.
   a. Desde la ventana Centro de Control, amplíe el árbol de objetos hasta encontrar la carpeta **Vistas** en la que están las vistas de la base de datos que utiliza para operaciones espaciales.
   b. Pulse la carpeta **Vistas**. Se mostrarán las vistas en el panel de contenido de la parte derecha de la ventana.
   c. Pulse con el botón derecho del ratón la vista que desee y pulse **Spatial Extender —> Capas Espaciales** en el menú emergente. Se abrirá la ventana Capas Espaciales.
   d. En la ventana Capas Espaciales, pulse **Crear**. Se abrirá la ventana Crear Capa Espacial.
2. Utilice el recuadro **Columna de capa** para especificar la columna que desea registrar como una capa.
3. En el campo **Capa espacial implícita**, especifique el nombre de la columna de tabla en la que se basa la columna de vista seleccionada. Esta columna de tabla ya debe estar registrada como una capa.
4. Pulse **Bien** para registrar la columna de vista especificada como una capa.

Nota: Para eliminar una tabla que contiene una columna que se ha registrado como capa, primero debe realizar una o más de las acciones siguientes:

- Cancelar el registro de la capa.
- Si la capa tiene un índice espacial, suprimir este índice.
- Si se ha definido una columna de vista sobre la columna de tabla, y si esta columna de vista se ha registrado como capa, cancelar también el registro de esta capa.

Capítulo 5. Cómo llenar columnas espaciales

Después de registrar las columnas espaciales como capas, está listo para suministrarles datos espaciales. Tal como se indica en el tema "De dónde proceden los datos espaciales" en la página 6, hay varias formas de suministrar estos datos: utilizando una función, denominada geocodificador, para que obtenga los datos a partir de datos de atributo, utilizando otras funciones para obtener datos a partir de otros datos espaciales o importando los datos desde archivos. Este capítulo:

- Contiene información sobre la geocodificación y explica cómo utilizar el Centro de Control para geocodificar datos de atributo en la modalidad de proceso por lotes
- Explica cómo importar y exportar datos y cómo utilizar el Centro de Control para importar datos a su GIS y exportarlos de su GIS

Para obtener más información sobre funciones que permiten obtener nuevos datos espaciales a partir de datos espaciales existentes, consulte la sección "Funciones que generan geometrías nuevas a partir de geometrías existentes" en la página 188.

Utilización de geocodificadores

Esta sección describe el proceso de geocodificar y explica cómo ejecutar un geocodificador en la modalidad de proceso por lotes desde el Centro de Control.

Acerca de la geocodificación

Tal como se indicó anteriormente, DB2 Spatial Extender utiliza una función, denominada *geocodificador*, para realizar dos actividades: convertir direcciones en datos espaciales y colocar estos datos espaciales en columnas de tabla. Esta sección explica las diferencias básicas entre geocodificadores y sus fuentes. También describe las dos modalidades en que se puede ejecutar un geocodificador y contiene una introducción sobre los factores a tener en cuenta cuando se tiene intención de utilizar un geocodificador.

Con Spatial Extender, puede:

- Utilizar el geocodificador por omisión que se suministra con Spatial Extender.
- Incorporar geocodificadores desarrollados por otros proveedores.
- Incorporar sus propios geocodificadores.

El geocodificador por omisión geocodifica direcciones de Estados Unidos y permite convertirlas en datos para ST_Point. Si necesita almacenar otros tipos de datos espaciales, puede incorporar un geocodificador que genere dichos datos. Si necesita datos espaciales que representen lugares de fuera de Estados Unidos, o lugares que no tienen dirección (por ejemplo, tierras de labranza que se distinguen por la composición del suelo) puede incorporar un geocodificador de acuerdo con esas necesidades.

Para poder incorporar un geocodificador, este debe estar registrado. Los usuarios y proveedores pueden registrar sus geocodificadores con el procedimiento almacenado db2gse.gse_register_gc. No se pueden registrar desde el Centro de Control. Para obtener información sobre db2gse.gse_register_gc, consulte la sección "db2gse.gse_register_gc" en la página 108. Para obtener información general sobre cómo utilizar los procedimientos almacenados de Spatial Extender, consulte el "Capítulo 9. Procedimientos almacenados" en la página 83.

Un geocodificador se puede ejecutar en dos modalidades:

- En *modalidad de proceso por lotes* intenta, en una sola operación, convertir todos los datos fuente existentes correspondientes a una columna espacial en datos espaciales y llenar la columna con dichos datos. Puede iniciar esta operación desde la ventana Ejecutar Geocodificador. También puede iniciarla en un programa de aplicación, codificando el programa para que llame al procedimiento almacenado db2gse.gse_run_gc.
- En *modalidad incremental*, un geocodificador convierte datos cuando estos se insertan o actualizan en una tabla, colocando los valores espaciales resultantes en una columna a fin de mantener la columna actualizada. Se activa mediante activadores de inserción y actualización que puede solicitar desde la ventana Crear capa espacial. También puede solicitarlos en un programa de aplicación, codificando el programa para que llame al procedimiento almacenado db2gse.gse_enable_autogc.

  La geocodificación incremental también recibe el nombre de *geocodificación automática*.

Cuando tenga intención de utilizar un geocodificador, tenga en cuenta los siguientes factores:

1. Cuando utilice el Centro de Control, generalmente utilizará la ventana Crear capas espaciales antes de utilizar la ventana Ejecutar Geocodificador. Esto significa que puede indicar a Spatial Extender que configure activadores para la geocodificación incremental antes de iniciar la geocodificación de proceso por lotes. Por lo tanto, es posible que la geocodificación incremental preceda a la geocodificación de proceso por lotes. Al procesar todos los datos fuente en la modalidad de proceso por lotes, el geocodificador geocodificará los mismos datos utilizados en la modalidad incremental. Esta redundancia no causará duplicaciones

(cuando los datos espaciales se producen dos veces, el segundo grupo de datos prevalece sobre el primero). Sin embargo, esto puede degradar el rendimiento. Un modo de evitarlo consiste en diferir la configuración de activadores hasta que termine la geocodificación de proceso por lotes.

2. Si los activadores están listos cuando el usuario esté preparado para geocodificar en la modalidad de proceso por lotes, se recomienda desactivarlos hasta que termine la geocodificación de proceso por lotes. Puede desactivarlos desde la ventana Ejecutar Geocodificador o en un programa de aplicación, codificando el programa para que llame al procedimiento almacenado db2gse.gse_disable_autogc. Si utiliza la ventana Ejecutar Geocodificador, Spatial Extender los vuelve a activar automáticamente cuando termina la geocodificación. Si utiliza el procedimiento almacenado db2gse.gse_disable_autogc, los puede volver a activar llamando al procedimiento almacenado db2gse.gse_enable_autogc.
3. Si desea ejecutar un geocodificador en la modalidad de proceso por lotes para llenar una columna espacial que tiene un índice, inhabilite o descarte antes el índice. De lo contrario, si el índice permanece operativo mientras se ejecuta el geocodificador, el rendimiento se degradará significativamente. Si utiliza el Centro de Control, puede inhabilitar el índice desde la ventana Ejecutar Geocodificador. Spatial Extender vuelve a habilitar el índice automáticamente cuando termina la geocodificación. Si utiliza un programa de aplicación, puede descartar el índice con la sentencia SQL DROP. Si lo hace, asegúrese de anotar los parámetros del índice para poder volver a generarlo una vez terminada la geocodificación de proceso por lotes.
4. Cuando el geocodificador lee un registro de datos fuente, intenta hacer coincidir dicho registro con su correspondiente en los datos de referencia. La coincidencia debe tener cierto grado de precisión (denominado *precisión*) para que el geocodificador procese el registro. Por ejemplo, una precisión de 85 significa que la coincidencia entre un registro fuente y su correspondiente en los datos de referencia debe tener una precisión de al menos un 85 por ciento para que se procese el registro fuente.

   El usuario especifica la precisión. Observe que es posible que tenga que ajustarla. Por ejemplo, supongamos que la precisión es 100. Si muchos registros fuente contienen direcciones que son más recientes que los datos de referencia, será imposible obtener una precisión del 100 por ciento en la comparación entre estos registros y los datos de referencia. Como resultado, el geocodificador rechazará estos registros. En resumen, si un geocodificador genera datos espaciales que parecen insuficientes o poco precisos, puede resolver este problema cambiando la precisión y volviendo a ejecutar el geocodificador.

Existen varias formas de controlar el número o rango de registros que un geocodificador procesa antes de emitirse una operación de confirmación:

Método 1

Un geocodificador puede geocodificar direcciones en un número específico de registros antes de cada confirmación. Este método permite al usuario gestionar el tamaño de las unidades de trabajo de una forma precisa. Sin embargo, este método comporta una actividad del sistema notablemente mayor que los otros métodos descritos aquí.

Para iniciar el Método 1, especifique el número de registros que se deben procesar antes de cada confirmación. Si utiliza el Centro de Control, puede especificar este número con el selector cíclico **Ámbito de confirmación** en la ventana Ejecutar Geocodificador. Si está escribiendo un programa de aplicación, asigne este número al parámetro commitScope del procedimiento almacenado db2gse.gse_run_gc.

Método 2

Un geocodificador puede geocodificar direcciones en todos los registros de una tabla antes de emitirse una confirmación. Con este método, el geocodificador procesa los registros en una forma similar a una operación de proceso por lotes; comporta una menor actividad por registro que el Método 1. Sin embargo, el usuario tiene menos control sobre el tamaño de la unidad de trabajo que en el Método 1. En consecuencia, no se puede controlar cuántos bloqueos se deben mantener ni cuántas entradas deben hacerse en el archivo de anotaciones durante el funcionamiento del geocodificador. Además, si el geocodificador encuentra un error que precisa que se haga una cancelación de cambios, es necesario ejecutar de nuevo el geocodificador para todos los registros. El consiguiente consumo de recursos puede ser alto si la tabla es muy grande y el error y la cancelación de cambios se producen una vez procesados la mayoría de los registros.

Para iniciar el Método 2 desde el Centro de Control, establezca en 0 el selector cíclico **Ámbito de confirmación** en la ventana Ejecutar Geocodificador. Si está escribiendo un programa de aplicación, asigne el valor 0 al parámetro commitScope del procedimiento almacenado db2gse.gse_run_gc.

Método 3

Un geocodificador puede geocodificar direcciones en un subconjunto de registros de una tabla antes de cada confirmación. Luego, puede geocodificar direcciones en un segundo subconjunto de registros y, si es necesario, en un tercero, un cuarto, etc. El geocodificador procesa los registros de cada subconjunto en una forma similar a una operación de proceso por lotes; este método comporta una menor actividad por registro que el Método 1. Sin embargo, al igual que el Método 2 y el Método 3, el usuario no tiene un

control directo sobre el tamaño de la unidad de trabajo. Además, es necesario preparar y ejecutar el geocodificador varias veces (una vez para cada subconjunto de registros).

Para definir un subconjunto de registros desde el Centro de Control:

- En el cuadro **Cláusula Where** de la ventana Ejecutar Geocodificador, codifique una cláusula WHERE de SELECT que especifique el rango de registros a procesar.
- Establezca en 0 el selector cíclico **Ámbito de confirmación** en la ventana Ejecutar Geocodificador.

Para definir un subconjunto de registros en un programa de aplicación, codifique el procedimiento almacenado db2gse.gse_run_gc de esta manera:

- Utilice el parámetro whereClause para especificar el rango de registros del subconjunto.
- Asigne el valor 0 al parámetro commitScope.

Ejecución del geocodificador en la modalidad de proceso por lotes

Esta sección contiene una visión general de los pasos a seguir para ejecutar un geocodificador en la modalidad de proceso por lotes desde el Centro de Control. La visión general va seguida de detalles sobre cómo realizar cada paso.

Para saber qué autorización necesita para ejecutar un geocodificador en la modalidad de proceso por lotes, consulte la sección "Autorización" en la página 118.

Visión general de los pasos a seguir para ejecutar un geocodificador en la modalidad de proceso por lotes:

1. Abra la ventana Ejecutar Geocodificador.
2. Indique qué geocodificador desea utilizar.
3. Inhabilite los objetos que puedan dificultar el rendimiento del geocodificador.
4. Especifique el número de registros a geocodificar antes de que DB2 emita un punto de compromiso.
5. Indique el modo en que desea ejecutar el geocodificador.
6. Indique a Spatial Extender que ejecute el geocodificador.

Pasos detallados para ejecutar un geocodificador en la modalidad de proceso por lotes:

1. Abra la ventana Ejecutar Geocodificador.

a. Desde la ventana Centro de Control, amplíe el árbol de objetos hasta encontrar la carpeta **Tablas** en su base de datos habilitada para operaciones espaciales.
b. Pulse la carpeta **Tablas**. Se mostrarán las tablas en el panel de contenido de la parte derecha de la ventana.
c. Pulse con el botón derecho del ratón la tabla que desee en el panel de contenido y pulse **Capas espaciales** en el menú emergente. Se abrirá la ventana Capas Espaciales.
d. En la ventana Capas espaciales:
   1) Seleccione la capa que está definida en la columna que desea llenar.
   2) Pulse el botón **Ejecutar Geocodificador**. Se abrirá la ventana Ejecutar Geocodificador.

2. Si desea utilizar el geocodificador por omisión, no modifique el recuadro **Nombre**, que muestra el nombre del geocodificador por omisión. Si no es así, utilice el recuadro para seleccionar el geocodificador que desee.
3. Inhabilite los objetos que puedan dificultar el rendimiento del geocodificador:
   - Si la columna que desea llenar tiene un índice, seleccione el recuadro **Inhabilitar temporalmente los índices espaciales durante el proceso de geocodificación**.
   - Si los activadores se han definido para que activen la geocodificación incremental para esta columna, seleccione el recuadro **Inhabilitar temporalmente activadores espaciales durante el proceso de geocodificación**.

     El índice y los activadores se volverán a habilitar automáticamente cuando pulse **Bien** en la ventana Ejecutar Geocodificador.
4. Utilice el selector cíclico **Ámbito de confirmación** para especificar el número de registros a geocodificar antes de que DB2 emita una confirmación. Por ejemplo, si desea que DB2 confirme 100 registros geocodificados cada vez, especifique el número 100.

   Sugerencia: Si desea que DB2 emita una confirmación sólo después de que se hayan procesado todos los registros, especifique cero.
5. Utilice los campos del recuadro de grupo **Parámetros de geocodificador** para indicar el modo en que desea ejecutar el geocodificador:
   - Utilice el selector cíclico **Nivel de precisión** para especificar, en términos de porcentaje, la precisión que debe tener una coincidencia entre los registros fuente y sus correspondientes en los datos de referencia. Para obtener más información sobre la precisión, consulte la sección "Acerca de la geocodificación" en la página 53.

- Si utiliza un geocodificador suministrado por un proveedor y desea utilizar las propiedades a las que da soporte, utilice el recuadro **Propiedades** para definir dichas propiedades.
- Si desea geocodificar únicamente un subconjunto de filas de la tabla seleccionada, utilice el recuadro **Cláusula WHERE** para codificar una cláusula SELECT WHERE que especificará los criterios para las filas que desea. Esta cláusula puede hacer referencia a cualquiera de las columnas de la tabla.

  Escriba únicamente los criterios. Omita la palabra clave WHERE. Por ejemplo, si la tabla tiene una columna STATE y desea geocodificar únicamente las filas que contienen el valor MA en esta columna, escriba:

  ```
  STATE='MA'
  ```

6. Pulse **Bien** para ejecutar el geocodificador.

Importación y exportación de datos

Esta sección describe el proceso de importar y exportar datos y explica cómo utilizar el Centro de Control para:

- Importar datos de un archivo de intercambio de datos a una tabla nueva o existente
- Importar datos de un archivo de intercambio de datos a una tabla existente
- Exportar datos de una tabla a un archivo de intercambio de datos

Acerca de la importación y la exportación

Esta sección lista las razones para importar y exportar datos espaciales. También describe los archivos de intercambio de datos que sirven como interfaces entre las fuentes de la exportación y los destinos de la importación.

Puede utilizar Spatial Extender para importar datos espaciales de archivos de intercambio de datos y para exportar datos a los mismos. Supongamos los casos siguientes:

- Su GIS contiene datos espaciales que representan oficinas, clientes y otros elementos referentes a su empresa. Desea complementar estos datos con datos espaciales que representen el entorno cultural de su organización: ciudades, calles, puntos de interés, etc. Los datos que desea pueden obtenerse de un proveedor de mapas. Puede utilizar Spatial Extender para importar estos datos desde un archivo de intercambio de datos que suministra el proveedor.
- Desea migrar datos espaciales desde un sistema Oracle a su GIS de Spatial Extender. Para ello utiliza un programa de utilidad de Oracle que carga los datos en un archivo de intercambio de datos. Luego utiliza Spatial Extender para importar los datos de este archivo a la base de datos que tiene habilitada para operaciones espaciales.

- Desea utilizar un explorador de GIS para mostrar a los clientes presentaciones visuales de información espacial. El explorador sólo necesita archivos con los que trabajar; no necesita estar conectado a una base de datos. Podría utilizar Spatial Extender para exportar los datos a un archivo de intercambio de datos y luego utilizar un programa de utilidad de explorador para cargar los datos en el explorador.

El Centro de Control da soporte a dos tipos de archivos de intercambio de datos para Spatial Extender: archivos de forma y archivos de transferencia ESRI_SDE. Los archivos de forma se suelen utilizar para importar datos que se originan en sistemas de archivos y para exportar datos a archivos que se deben cargar en sistemas de archivos. Los archivos de transferencia de datos ESRI_SDE se suelen utilizar para importar datos que se originan en bases de datos ESRI.

Importación de datos a una tabla nueva o existente desde el nivel de la base de datos

Esta sección da una visión general de los pasos para importar datos desde un archivo de formas o archivo de transferencia ESRI_SDE a una tabla nueva o existente utilizando la carpeta **Bases de datos** del Centro de Control. La visión general va seguida de detalles sobre cómo realizar cada paso.

Cuando importa un conjunto de representaciones de formas ESRI, recibe al menos dos archivos. Todos los archivos tienen el mismo nombre, pero extensiones diferentes. Por ejemplo, las extensiones de los dos archivos que recibe siempre son .shp y .shx.

Para recibir los archivos correspondientes a un conjunto de representaciones de formas, escriba el nombre compartido de los archivos en el campo **Nombre de archivo** de la ventana Importar Datos Espaciales. No especifique una extensión de archivo. De esta forma, se asegura la importación de todos los archivos que necesita: el archivo .shp, el archivo .shx y cualquier otro que pudiera incluirse.

Por ejemplo, suponga que un conjunto de representaciones de formas ESRI está almacenado en los archivos llamados Lakes.shp y Lakes.shx. Cuando importe estas representaciones, escribirá sólo ″Lakes″ en el campo Nombre de Archivo.

Los archivos de transferencia SDE tienen nombres, pero carecen de extensión. Por tanto, cuando importe un archivo de transferencia SDE, escriba su nombre, pero sin extensión, en el campo Nombre de Archivo. Similarmente, en el campo **Archivo de excepción** de las ventanas Importar Datos Espaciales, no escriba la extensión del archivo que desea especificar. Escriba sólo el nombre del archivo.

Para determinar qué autorización es necesaria para importar datos de forma, consulte la sección "Autorización" en la página 105. Para determinar qué autorización es necesaria para importar datos ESRI_SDE, consulte la sección "Autorización" en la página 103.

Visión general de los pasos para importar datos a una tabla nueva o existente:

1. Abra la ventana Importar Datos Espaciales.
2. Especifique la vía de acceso, el nombre y el formato del archivo que contiene los datos que desea importar.
3. Especifique el número de registros a importar antes de cada confirmación.
4. Si desea importar datos espaciales a una tabla que se tiene que crear, suministre un nombre para esta tabla y un nombre para la columna a la que van destinados los datos. Si va a importar datos espaciales a una tabla existente, indique a qué columna van destinados los datos.
5. Especifique qué sistema de referencias espaciales se debe asociar a los datos.
6. Designe un archivo para que recopile los registros que no se puedan importar.
7. Indique a Spatial Extender que importe los datos y, si ha definido una tabla en esta ventana, que cree la tabla y que registre la columna a la que van destinados los datos como una capa.

Detalles de los pasos a seguir para importar datos a una tabla nueva o existente:

1. Abra la ventana Importar Datos Espaciales.
   a. Desde la ventana Centro de Control, amplíe el árbol de objetos hasta encontrar la carpeta **Bases de datos** bajo el servidor en el que está ejecutando Spatial Extender.
   b. Pulse la carpeta **Bases de datos**. Se muestran las bases de datos en el panel de contenido en la parte derecha de la ventana.
   c. Pulse con el botón derecho del ratón la base de datos a la que desea importar datos y pulse **Spatial Extender —> Importar Datos Espaciales** en el menú emergente. Se abrirá la ventana Importar Datos Espaciales.
2. Especifique la vía de acceso, el nombre y el formato del archivo que contiene los datos que desea importar:
   a. Utilice el campo **Nombre de archivo** para especificar la vía de acceso y el nombre.
   b. Utilice el recuadro **Formato de archivo** para especificar el formato. El formato puede ser:

      Forma Éste es el valor por omisión.

ESRI_SDE

Si especifica este formato, el campo **Nombre de referencia espacial** adopta por omisión el nombre del sistema de referencias espaciales asociado a este formato.

3. Utilice el campo **Ámbito de confirmación** para especificar el número de registros que desea importar antes de cada confirmación. Por ejemplo, para solicitar a DB2 que confirme 100 registros cada vez, especifique el número 100.

   Sugerencia: Si desea que DB2 emita una confirmación sólo después de que se hayan procesado todos los registros, especifique cero.

4. Especifique la tabla y columna a las que van destinados los datos.
   a. Utilice el recuadro **Esquema de capa** para especificar el esquema correspondiente a la tabla en la que se van a importar los datos.
   b. Especifique la tabla y la columna:
      - Si la tabla aún no existe:
        1) En el campo **Tabla de capa**, escriba un nombre para la tabla.
        2) En el campo **Columna de capa**, escriba un nombre para la columna que va a contener los datos importados. Spatial Extender registrará automáticamente esta columna como una capa.
      - Si la tabla ya existe:
        1) En el campo **Tabla de capa**, especifique la tabla. Ya debe contener la columna en la que desea colocar los datos importados. Además, esta columna ya debe estar registrada como una capa.
        2) En el campo **Columna de capa**, especifique el nombre de la columna a la que van destinados los datos importados.
5. En el campo **Nombre de referencia espacial**, escriba o seleccione el sistema de referencias espaciales que se debe asociar a estos datos. (Si los datos van a proceder de un archivo de transferencia ESRI_SDE, el nombre del sistema de referencias espaciales asociado se muestra automáticamente en el campo.)
6. En el campo **Archivo de excepciones**, especifique la vía de acceso y el nombre correspondiente a un archivo nuevo en el que se recopilarán los registros que no se puedan importar. Luego puede arreglar estos registros e importarlos desde este archivo.

   Spatial Extender creará este archivo; no especifique uno que ya exista.
7. Pulse **Bien** para importar los datos. Además, si ha suministrado un nombre para una tabla que aún no existe, esta tabla se creará y la columna a la que van destinados los datos se registrará como una capa. También se creará el archivo de excepciones especificado.

Importación de datos a una tabla existente desde el nivel de tabla

Esta sección da una visión general de los pasos para importar datos desde un archivo de formas o archivo de transferencia ESRI_SDE a una tabla existente utilizando la carpeta **Tablas** del Centro de Control. La visión general va seguida de detalles sobre cómo realizar cada paso.

Para saber qué autorización se necesita para importar datos de forma, consulte la sección "Autorización" en la página 105. Para saber qué autorización se necesita para importar datos ESRI_SDE, consulte la sección "Autorización" en la página 103.

Visión general de los pasos a seguir para importar datos a una tabla existente:

1. Abra la ventana Importar Datos Espaciales.
2. Especifique la vía de acceso y el nombre del archivo que contiene los datos que desea importar.
3. Especifique el número de registros a importar antes de cada confirmación.
4. Especifique la columna que debe contener los datos espaciales importados.
5. Especifique qué sistema de referencias espaciales se debe asociar a estos datos.
6. Designe un archivo para que recopile los registros que no se puedan importar.
7. Indique a Spatial Extender que importe los datos y, si ha especificado una columna que aún no se ha creado, que la cree y que la registre como una capa.

Detalles de los pasos a seguir para importar datos a una tabla existente:

1. Abra la ventana Importar Datos Espaciales.
   a. Desde la ventana Centro de Control, amplíe el árbol de objetos hasta encontrar la carpeta **Tablas** correspondiente a la base de datos a la que desea importar datos.
   b. Pulse la carpeta **Tablas**. Se mostrarán las tablas en el panel de contenido de la parte derecha de la ventana.
   c. Pulse con el botón derecho del ratón la tabla en la que desea importar datos y pulse **Spatial Extender —> Importar Datos Espaciales** en el menú emergente. Se abrirá la ventana Importar Datos Espaciales.
2. En el recuadro **Nombre de archivo**, especifique la vía de acceso y el nombre del archivo que contiene los datos a importar.
3. Utilice el recuadro **Ámbito de confirmación** para especificar el número de registros que desea importar antes de cada confirmación. Por ejemplo, para solicitar a DB2 que confirme 100 registros cada vez, especifique el número 100.

Sugerencia: Si desea que DB2 emita una confirmación sólo después de que se hayan procesado todos los registros, especifique cero.

4. Especifique la columna que debe contener los datos espaciales importados.
   - Si la columna aún no existe en la tabla, utilice el recuadro **Columna de capa** para escribir un nombre para la columna.
   - Si la columna ya existe, utilice el recuadro **Columna de capa** para seleccionar o escribir el nombre de la columna.
5. Utilice el recuadro **Nombre de referencia espacial** para especificar qué sistema de referencias espaciales se debe asociar a los datos importados.
   - Si está añadiendo una columna a una tabla, escriba o seleccione el nombre del sistema de referencias espaciales.
   - Si los datos importados van destinados a una columna existente, no modifique el recuadro **Nombre de referencia espacial**. Este muestra el nombre del sistema de referencias espaciales por omisión.
6. En el campo **Archivo de excepciones**, especifique la vía de acceso y el nombre correspondiente a un archivo nuevo en el que se recopilarán los registros que no se puedan importar. Luego puede arreglar estos registros e importarlos desde este archivo.

   Spatial Extender creará este archivo; no especifique uno que ya exista.
7. Pulse **Bien** para importar los datos. Además, si ha especificado una columna que aún no existe, esta columna se creará y se registrará como una capa. También se creará el archivo de excepciones especificado.

Exportación de datos a un archivo de formas

Esta sección contiene una visión general de los pasos a seguir para exportar datos a un archivo de formas. La visión general va seguida de detalles sobre cómo realizar cada paso.

Para saber qué autorización se necesita para llevar a cabo estos pasos, consulte la sección "Autorización" en la página 101.

Visión general de los pasos a seguir para exportar datos de un archivo de formas:

1. Abra la ventana Exportar datos espaciales.
2. Especifique la columna que contiene los datos espaciales que se deben exportar.
3. Si desea exportar un subconjunto de filas de datos, identifique este subconjunto ante Spatial Extender.
4. Especifique la vía de acceso y el nombre del archivo al que va a exportar los datos.
5. Indique a Spatial Extender que exporte los datos.

Detalles de los pasos a seguir para exportar datos a un archivo de formas:

1. Abra la ventana Exportar datos espaciales.
   a. En la ventana Centro de Control, amplíe el árbol de objetos hasta encontrar la carpeta **Tablas** o **Vistas** de la base de datos que contiene datos espaciales:
   b. Pulse la carpeta **Tablas** o **Vistas**. Se mostrarán las tablas o vistas en el panel de contenido de la parte derecha de la ventana.
   c. Pulse con el botón derecho del ratón la tabla o vista que contiene los datos a exportar y pulse **Spatial Extender** —> **Exportar datos espaciales** en el menú emergente. Se abrirá la ventana Exportar datos espaciales.
2. En el campo **Columna de capa**, especifique el nombre de la columna que contiene los datos espaciales a exportar.
3. Si desea exportar un subconjunto de filas de la tabla, utilice el recuadro **Cláusula WHERE** para escribir una cláusula WHERE que especifique los criterios correspondientes a las filas que desea. En esta cláusula sólo puede hacer referencia a columnas de la tabla o vista de la que está exportando datos.

   Escriba únicamente los criterios. Omita la palabra clave WHERE. Por ejemplo, si la tabla o vista tiene una columna STATE y desea geocodificar únicamente las filas que contienen el valor MA en esta columna, escriba:

   ```
   STATE='MA'
   ```
4. En el campo **Nombre de archivo**, especifique la vía de acceso y el nombre del archivo al que está exportando datos.
5. Pulse **Bien** para exportar los datos.

Capítulo 6. Creación de índices espaciales

Este capítulo explica cómo utilizar el Centro de Control para crear un índice para datos espaciales.

Después de llenar con sus datos las columnas espaciales, ya está listo para crear un índice espacial. Las típicas estructuras de índice, como un árbol B, realizan clasificaciones lineales, de una dimensión, de los datos de una tabla. Los datos de una tabla habilitados para operaciones espaciales no se almacenan como una sola entrada, sino que tienen dos dimensiones. Por ejemplo, las geometrías espaciales, como un polígono, constan de varios valores de coordenadas en una columna o capa espacial. Puesto que un índice tipo árbol B no puede manejar tipos de datos espaciales, Spatial Extender ha creado su propia tecnología de indexación denominada *índice de cuadrícula*. El índice de cuadrícula se basa en el índice tipo árbol B, pero mejorado para manejar datos de dos dimensiones y realizar indexación de columnas espaciales. El índice de cuadrícula da soporte a tres capas y está diseñado para ofrecer un buen rendimiento en una amplia gama de objetos, tamaños y distribuciones de datos. Para obtener más información sobre índices espaciales, consulte el "Capítulo 12. Índices espaciales" en la página 149.

Para saber qué autorización se necesita para crear un índice espacial, consulte la sección "Autorización" en la página 96.

Utilización del Centro de Control para crear un índice espacial

Para crear un índice espacial mediante el Centro de Control:

1. En el árbol de objetos, seleccione la carpeta **Tablas**. Todas las tablas existentes se mostrarán en el panel de contenido.
2. En el panel de contenido, pulse con el botón derecho del ratón la tabla para la que desea crear un índice y pulse **Spatial Extender —> Índices espaciales** en el menú emergente. Se abrirá la ventana Índices espaciales.
3. En la ventana Índices espaciales, pulse **Crear**. Se abrirá la ventana Crear índice espacial.
4. En el campo **Nombre**, escriba el nombre del nuevo índice espacial que desea crear.

   Nota: Spatial Extender no permite que el usuario seleccione un nombre de esquema para un índice. Spatial Extender añade automáticamente el nombre de esquema y crea un nombre totalmente calificado.

5. En el campo **Columna de capa**, seleccione la capa para la que está creando un índice. Una capa puede tener un solo índice espacial.

   Una capa es una columna espacial definida o registrada ante Spatial Extender.

6. En los campos **Tamaño de cuadrícula**, escriba el valor de tamaño de cuadrícula que desea asignar a cada campo.

   Los niveles de cuadrícula, **Fino**, **Medio** y **Grueso**, se entran aumentando el tamaño de celda. Por lo tanto, el segundo nivel debe tener un tamaño de celda mayor que el primero, y el tercero uno mayor que el segundo.

   Para definir un tamaño de cuadrícula en el campo **Fino**, **Medio** o **Grueso**, pulse las teclas de flecha arriba y fecha abajo del teclado de la estación de trabajo. Pulse la tecla de flecha arriba para aumentar el tamaño en una décima de grado; pulse la tecla de flecha abajo para disminuir el tamaño en una décima de grado.

   No es necesario que especifique los 3 niveles de tamaño de cuadrícula.

Determinación de los tamaños de celda de cuadrícula

La determinación del tamaño de cuadrícula correcto se realiza mediante un proceso de prueba y error. Se recomienda definir un tamaño de celda en relación al tamaño aproximado del objeto que está indexando. Un tamaño de cuadrícula demasiado pequeño o demasiado grande puede reducir el rendimiento de las operaciones. Los tamaños demasiado pequeños afectan a la relación clave/objeto durante una búsqueda por índice. En este caso, se crean demasiadas claves y se devuelve un gran número de candidatos. Si el tamaño de cuadrícula es demasiado grande, la búsqueda inicial por índice devuelve un pequeño número de candidatos, pero el rendimiento puede disminuir durante la exploración final de la tabla.

Para obtener más información sobre cómo seleccionar tamaños de celda de cuadrícula y el número de niveles de cuadrícula, consulte la sección “Selección del tamaño de celda de cuadrícula” en la página 156.

Capítulo 7. Recuperación y análisis de información espacial

Después de construir índices espaciales, ya se pueden utilizar las tablas espaciales. Este capítulo trata temas relacionados con la recuperación y análisis de datos espaciales. Contiene una visión general de diversos métodos de recuperación y ofrece ejemplos de consultas de tablas que utilizan funciones espaciales.

Métodos de realizar análisis espaciales

Puede realizar análisis espaciales utilizando SQL y funciones espaciales con cualquiera de los siguientes entornos de programación:

- Un geoexplorador (por ejemplo, ArcExplorer de ESRI).

  Para obtener más información sobre la utilización de ArcExplorer, consulte el manual *Using ArcExplorer*, disponible en el sitio Web de ESRI en la dirección http://www.esri.com.
- Sentencias de SQL interactivo.

  Puede emitir sentencias de SQL interactivo desde el Centro de Mandatos de DB2, la ventana Mandato de DB2 o el procesador de línea de mandatos.
- Aplicaciones desarrolladas por el usuario (por ejemplo, ODBC, JDBC y SQL intercalado).

Creación de una consulta espacial

Esta sección explica cómo crear consultas espaciales que utilizan predicados y funciones espaciales.

Funciones espaciales y SQL

Spatial Extender incluye funciones que realizan varias operaciones sobre datos espaciales. Los ejemplos de esta sección le muestran cómo utilizar funciones espaciales para crear sus propias consultas espaciales.

La Tabla 3 contiene una lista de funciones espaciales y los tipos de operaciones que pueden realizar.

Tabla 3. Funciones y operaciones espaciales

Tipo de función	Ejemplo de operación
Cálculo	Calcular la distancia entre dos puntos
Comparación	Buscar todos los clientes ubicados en una zona de inundaciones

Tabla 3. Funciones y operaciones espaciales (continuación)

Tipo de función	Ejemplo de operación
Intercambio de datos	Convertir datos a formatos soportados
Transformación	Añadir un radio de cinco millas a un punto

Para obtener más información sobre funciones espaciales, consulte el "Capítulo 13. Geometrías y funciones espaciales asociadas" en la página 159 y el "Capítulo 14. Funciones espaciales para consultas SQL" en la página 199.

Ejemplo 1: Comparación

La siguiente consulta busca la distancia media de los clientes de cada centro comercial. Las funciones espaciales utilizadas en este ejemplo son ST_Distance y ST_Within.

```
SELECT s.id, AVG(db2gse.ST_Distance(c.location,s.location))
FROM customers c, stores s
WHERE db2gse.ST_Within(c.location,s.zone)=1
GROUP BY s.id
```

Ejemplo 2: Intercambio de datos

La siguiente consulta busca las ubicaciones de los clientes que viven en el área de la bahía de San Francisco. Las funciones espaciales utilizadas en este ejemplo son ST_AsText (intercambio de datos) y ST_Within. ST_AsText convierte los datos espaciales de la columna c.location en el formato OGC TEXT.

```
SELECT db2gse.ST_AsText(c.location,cordref(1))
FROM customers c
WHERE db2gse.ST_Within(c.location,:BayArea)=1
```

Ejemplo 3: Cálculo

La siguiente consulta busca todas las calles de más de 10,5 millas. La función espacial utilizada es ST_Length.

```
SELECT s.name,s.id
FROM street s
WHERE db2gse.ST_Length(s.path) > 10.5
```

Ejemplo 4: Transformación

Esta consulta busca los clientes que viven dentro de una zona con riesgo de inundación o a menos de 2 millas del límite de la zona con riesgo de inundación. Las funciones espaciales utilizadas en este ejemplo son ST_Buffer (transformación) y ST_Within. La variable :floodzone es una variable de lenguaje principal situada en un programa de SQL intercalado que está escrito en C/C++.

```
SELECT c.name,c.phoneNo,c.address
FROM customers c
WHERE db2gse.ST_Within(c.location,ST_Buffer(:floodzone,2))=1
```

Predicados espaciales y SQL

Un grupo especializado de funciones espaciales denominadas predicados espaciales permiten mejorar el rendimiento de las consultas. Los predicados espaciales, como ST_Overlaps, que compara dos polígonos para ver si se solapan, pueden resultar caros de ejecutar por requisitos tanto de tiempo como de memoria. Por lo tanto, las técnicas de optimización para minimizar el coste de ejecución resultan muy importantes. El optimizador de consultas de DB2 utiliza el índice espacial para mejorar el rendimiento de las consultas cuando el usuario utiliza predicados espaciales según las reglas que se describen más adelante en esta sección. Para obtener más información sobre predicados espaciales, consulte la sección "Funciones de predicado" en la página 175. Los predicados espaciales utilizados para aprovechar el índice espacial son:

- ST_Contains
- ST_Crosses
- ST_Disjoint
- ST_Distance
- ST_Envelope
- ST_Equals
- ST_Intersects
- ST_Overlaps
- ST_Touches
- ST_Within

Para ver una lista completa de todas las funciones y predicados espaciales, consulte el "Capítulo 14. Funciones espaciales para consultas SQL" en la página 199.

Reglas para el aprovechamiento del índice

Las siguientes reglas sirven para optimizar las consultas espaciales que utilizan predicados espaciales:

- El predicado se debe utilizar en la cláusula WHERE.
- El predicado debe quedar a la izquierda de la comparación. Por ejemplo:

  ```
  WHERE db2gse.ST_Within(c.location,:BayArea)=1
  ```

- Las comparaciones de igualdad deben utilizar la constante de entero 1.

  ```
  WHERE db2gse.ST_Within(c.location,:BayArea)=1
  ```

- Debe haber una columna espacial que se utilice en el predicado como el destino de la búsqueda y debe haber un índice espacial creado en dicha columna.

Ejemplos de aprovechamiento del índice

La Tabla 4 en la página 72 muestra formas correctas e incorrectas de crear consultas espaciales para aprovechar el índice espacial.

Tabla 4. Reglas para el aprovechamiento del índice

Consulta espacial	Regla violada

`SELECT *`
`FROM customers c`
`WHERE db2gse.ST_Within(c.location,:BayArea)=1`
	En este ejemplo no se viola ninguna condición.

`SELECT *`
`FROM customers c`
`WHERE db2gse.ST_Distance(c.location,:SanJose)<10`
	En este ejemplo no se viola ninguna condición.

`SELECT *`
`FROM customers c`
`WHERE db2gse.ST_Length(c.location)>10`
	ST_Length no es un predicado espacial.

`SELECT *`
`FROM customers c`
`WHERE 1=db2gse.ST_Within(c.location,:BayArea)`
	El predicado debe quedar a la izquierda de la comparación.

`SELECT *`
`FROM customers c`
`WHERE db2gse.ST_Within(c.location,:BayArea)=2`
	Las comparaciones de igualdad deben utilizar la constante de entero 1.

`SELECT *`
`FROM customers c`
`WHERE db2gse.ST_Within(:SanJose,:BayArea)=1`
	Debe haber una columna espacial que se utilice en el predicado como el destino de la búsqueda y debe haber un índice espacial creado en dicha columna. (SanJose y BayArea no son columnas espaciales y, por lo tanto, no pueden tener asociado un índice espacial.)

Capítulo 8. Escritura de aplicaciones para Spatial Extender

Este capítulo explica cómo utilizar el programa de ejemplo de Spatial Extender para escribir aplicaciones para trabajar con información espacial y personalizarla. Se incluyen los siguientes temas:

- Utilización del programa de ejemplo
- Los pasos del programa de ejemplo

Utilización del programa de ejemplo

El programa de ejemplo de Spatial Extender facilita la programación de aplicaciones. Con el programa de ejemplo, puede:

- Automatizar los procedimientos espaciales rutinarios
- Cortar y pegar código de ejemplo en sus propias aplicaciones
- Entender los pasos que suelen ser necesarios para crear y mantener una base de datos habilitada para operaciones espaciales

Utilice el programa de ejemplo para codificar tareas complejas para Spatial Extender, por ejemplo para escribir una aplicación que utilice la interfaz de base de datos para llamar a procedimientos almacenados de Spatial Extender. Desde el programa de ejemplo, puede copiar y personalizar sus aplicaciones. Si no está familiarizado con los pasos de programación a seguir para Spatial Extender, puede ejecutar el programa de ejemplo, que le enseña cada paso con detalle. Antes debe crear el programa de ejemplo. Para ello utilice el archivo makefile de ejemplo. Para ver instrucciones sobre cómo crear y ejecutar el programa de ejemplo, consulte “Verificación de la instalación” en la página 26.

Los pasos del programa de ejemplo

La Tabla 5 en la página 74 muestra los pasos del programa de ejemplo, los procedimientos almacenados asociados y una descripción de cada paso. Las funciones C para invocar los procedimientos almacenados se muestran en la columna Acción de la Tabla 5 en la página 74 y aparecen entre paréntesis. Para obtener más información sobre los procedimientos almacenados, consulte el “Capítulo 9. Procedimientos almacenados” en la página 83. El programa de ejemplo se basa en los casos prácticos descritos en el tema “Ejemplo: una compañía de seguros actualiza su GIS” en la página 12.

Tabla 5. Programa de ejemplo de Spatial Extender

Pasos del programa de ejemplo	Acción	Descripción
Habilitar/inhabilitar la base de datos espacial	1. Habilitar la base de datos espacial (gseEnableDB) 2. Inhabilitar la base de datos espacial (gseDisableDB) 3. Habilitar la base de datos espacial (gseEnableDB)	1. Este es el primer paso a seguir para poder utilizar Spatial Extender. Una base de datos habilitada para operaciones espaciales tiene una serie de tipos espaciales, una serie de funciones espaciales, una serie de predicados espaciales, un nuevo tipo de índice y una serie de vistas y tablas de administración. 2. Este paso se suele llevar a cabo cuando ha habilitado funciones espaciales para la base de datos equivocada. Cuando inhabilita una base de datos espacial, elimina un conjunto de tipos espaciales, un conjunto de funciones espaciales, un conjunto de predicados espaciales, un nuevo tipo de índice y un conjunto de vistas y tablas de administración. **Nota:** La inhabilitación de la base de datos fallará si hay objetos creados que dependen de los objetos creados por el procedimiento de habilitación de la base de datos. Por ejemplo, la creación de una tabla con una columna espacial del tipo ST_Point hará que falle la inhabilitación de la base de datos. Esto sucede porque la tabla depende del tipo ST_Point, el cual debe eliminar el procedimiento de inhabilitación de la base de datos. 3. Igual que 1.

Tabla 5. Programa de ejemplo de Spatial Extender (continuación)

Pasos del programa de ejemplo	Acción	Descripción
Registrar sistemas de referencias espaciales	1. Registrar el sistema de referencias espaciales para la columna LOCATION de la tabla CUSTOMERS (gseEnableSref) 2. Registrar el sistema de referencias espaciales para la columna LOCATION de la tabla OFFICES (gseEnableSref) 3. Desregistrar el sistema de referencias espaciales para la columna LOCATION de la tabla OFFICES (gseDisableSref) 4. Volver a registrar el sistema de referencias espaciales para las columnas ZONE de la tabla OFFICES (gseEnableSref)	1. Este paso define un nuevo sistema de referencias espaciales (SRS) cuya finalidad es interpretar los datos espaciales de la tabla CUSTOMERS. Un sistema de referencias espaciales incluye datos de geometría en un formato que se puede almacenar en una columna de una base de datos habilitada para operaciones espaciales. Después de que un SRS se registra ante una determinada capa, las coordenadas aplicables a dicha capa se pueden almacenar en la columna de la tabla CUSTOMERS asociada. 2. Este paso define un nuevo sistema de referencias espaciales (SRS) cuya finalidad es interpretar los datos espaciales de la capa OFFICES. Cada capa de tabla debe tener un SRS definido. Es posible que las capas de la tabla OFFICES necesiten un SRS asociado distinto del de la capa de la tabla CUSTOMERS. 3. Este paso sólo se lleva a cabo si el usuario especifica los parámetros de SRS incorrectos ante la capa o columna espacial. Cuando el usuario desregistra un SRS correspondiente a la capa de la tabla OFFICES, elimina la definición con sus parámetros asociados. 4. Este paso define un nuevo sistema de referencias espaciales (SRS) cuya finalidad es interpretar los datos espaciales de la capa OFFICES.

Tabla 5. Programa de ejemplo de Spatial Extender (continuación)

Pasos del programa de ejemplo	Acción	Descripción
Crear las tablas espaciales	1. Modificar la tabla CUSTOMERS, añadiendo la columna LOCATION (gseSetupTables) 2. Crear la tabla OFFICES (gseSetupTables)	1. La tabla CUSTOMERS representa datos comerciales que se han almacenado en la base de datos durante varios años. La sentencia ALTER TABLE añade una nueva columna (LOCATION) del tipo ST_Point. Esta columna se llenará geocodificando las columnas de dirección en un paso posterior. 2. La tabla OFFICES representa, entre otros datos, la zona de ventas correspondiente a cada oficina de una compañía de seguros. La tabla entera se llenará con los datos de atributo procedentes de una base de datos que no es de DB2 en un paso posterior. Este paso incluye la importación de datos de atributo en la tabla OFFICES procedentes del archivo SHAPE.
Registrar capas espaciales	1. Registrar la columna LOCATION en la tabla CUSTOMERS como una capa (gseRegisterLayer) 2. Cancelar el registro de la columna LOCATION en la tabla CUSTOMERS (gseUnregisterLayer) 3. Registrar la columna ZONE de la tabla OFFICES como capa (gseRegisterLayer)	Estos pasos registran las columnas LOCATION y ZONE como capas ante Spatial Extender. Para que los programas de utilidad de Spatial Extender (por ejemplo, el geocodificador) puedan llenar o acceder a una columna espacial, tiene que registrarla como una capa. Puede también cancelar el registro de una capa después de hacerla accesible a los programas de utilidad de Spatial Extender. La columna asociada seguirá existiendo después de cancelar su registro como una capa.

Tabla 5. Programa de ejemplo de Spatial Extender (continuación)

Pasos del programa de ejemplo	Acción	Descripción
Llenar las capas espaciales	1. Geocodificar los datos de dirección correspondientes a la columna LOCATION de la tabla CUSTOMERS (gseRunGC) 2. Cargar la tabla OFFICES utilizando la modalidad de agregación (gseImportShape) 3. Cargar la tabla HAZARD_ZONE utilizando la modalidad de creación (gseImportShape)	1. Este paso lleva a cabo una geocodificación de proceso por lotes invocando el programa de utilidad del geocodificador. La geocodificación de proceso por lotes se suele realizar cuando una parte significativa de la tabla se tiene que geocodificar o se tiene que volver a geocodificar. 2. Este paso carga la tabla OFFICES con datos espaciales existentes en el formato de un archivo SHAPE. Puesto que la tabla OFFICES existe y la capa OFFICES/ZONE está registrada, el programa de utilidad de carga agregará los nuevos registros a una tabla existente. 3. Este paso carga la capa HAZARD_ZONE con datos espaciales existentes en el formato de un archivo SHAPE. Puesto que ni la tabla ni la capa existen, el programa de utilidad creará la tabla y registrará la capa antes de que se carguen los datos.
Registrar el geocodificador	• Registrar el geocodificador si no es el geocodificador por omisión (gseRegisterGc) • Cancelar el registro del geocodificador que se pueda haber registrado (gseUnregisterGc) • Registrar el geocodificador si no es el geocodificador por omisión (gseRegisterGc)	

Tabla 5. Programa de ejemplo de Spatial Extender (continuación)

Pasos del programa de ejemplo	Acción	Descripción
Habilitar índices espaciales	1. Habilitar el índice espacial para la columna LOCATION de la tabla (gseEnableIdx) 2. Habilitar el índice espacial para la columna ZONE de la tabla OFFICES (gseEnableIdx) 3. Habilitar el índice espacial para la columna LOCATION de la tabla OFFICES (gseEnableIdx) 4. Habilitar el índice espacial para la columna BOUNDRY de la tabla HAZARD_ZONE (gseEnableIdx)	Estos pasos habilitan el índice espacial para las tablas CUSTOMERS, OFFICES y HAZARD_ZONE.
Habilitar la geocodificación automática	1. Habilitar la geocodificación automática para las columnas LOCATION y ADDRESS de la tabla CUSTOMERS (gseEnableAutoGC)	Esta paso activa la invocación automática del geocodificador. La geocodificación automática permite que las columnas LOCATION y ADDRESS de la tabla CUSTOMERS estén sincronizadas entre sí para futuras operaciones de inserción y actualización.
Insertar/actualizar /suprimir la tabla CUSTOMERS	1. Insertar algunos registros con una calle diferente (gseInsDelUpd) 2. Actualizar algunos registros con una dirección nueva (gseInsDelUpd) 3. Suprimir todos los registros de la tabla (gseInsDelUpd)	Estos pasos muestran cómo realizar una inserción, actualización y supresión en la columna LOCATION de la tabla CUSTOMERS. Una vez habilitada la geocodificación automática, la información procedente de la columna ADDRESS se geocodifica automáticamente cuando se inserta o actualiza en la columna LOCATION. Este proceso se ha habilitado en el paso anterior.
Inhabilitar la geocodificación automática	1. Inhabilitar la geocodificación automática para la capa CUSTOMERS (gseDisableAutoGC) 2. Inhabilitar el índice espacial para la capa CUSTOMERS (gseDisableIdxCustomersLayer)	Estos pasos inhabilitan la invocación automática del geocodificador y del índice espacial como preparación para el siguiente paso (el siguiente paso incluye una nueva geocodificación de la tabla CUSTOMERS entera). Si está cargando gran cantidad de geodatos, se recomienda inhabilitar el índice espacial antes de cargar los datos y luego habilitarlo una vez finalizada la carga.

Tabla 5. Programa de ejemplo de Spatial Extender (continuación)

Pasos del programa de ejemplo	Acción	Descripción
Volver a geocodificar la tabla CUSTOMERS	1. Geocodificar de nuevo la capa CUSTOMERS con un nivel de precisión más bajo – 90% en lugar de 100% (gseRunGC) 2. Volver a habilitar el índice espacial para la capa CUSTOMERS (gseEnableIdx) 3. Volver a habilitar la geocodificación automática con un nivel de precisión más bajo – 90% en lugar de 100% (gseEnableAutoGC)	Estos pasos ejecutan de nuevo el geocodificador en modalidad de proceso por lotes, vuelven a habilitar la geocodificación automática con un nuevo nivel de precisión y vuelven a habilitar el índice espacial y la geocodificación automática. Esta acción se recomienda cuando el administrador espacial advierte un alto nivel de errores en el proceso de geocodificación. Si el nivel de precisión se define en 100%, es posible que no pueda geocodificar una dirección porque no encuentra una dirección correspondiente en los datos de referencia. Al reducir el nivel de precisión, el geocodificador tiene más probabilidades de encontrar datos coincidentes. Una vez se ha vuelto a geocodificar la tabla en modalidad de proceso por lotes, tanto la geocodificación automática como el índice espacial se vuelven a habilitar para facilitar el mantenimiento incremental del índice espacial y de la columna espacial para futuras operaciones de inserción y actualización.
Crear una vista y registrar sus columnas espaciales como capas de vista	1. Crear una vista, HIGHRISK_CUSTOMERS, basada en la unión de la tabla CUSTOMERS y la tabla HAZARD_ZONE (gseCreateView) 2. Registrar las columnas espaciales de la vista como capas (gseRegisterLayer)	Estos pasos crean una vista y registran sus columnas espaciales como capas de vista.

Tabla 5. Programa de ejemplo de Spatial Extender (continuación)

Pasos del programa de ejemplo	Acción	Descripción
Realizar análisis espacial	1. Buscar la distancia media de los clientes de cada oficina (ST_Within, ST_Distance) 2. Buscar el ingreso y la prima media por cliente para cada oficina (ST_Within) 3. Buscar clientes que no están cubiertos por ninguna oficina existente (ST_Within) 4. Buscar el número de zonas peligrosas que solapa cada zona de oficinas (ST_Overlaps) 5. Buscar la oficina más cercana a la ubicación de un determinado cliente, suponiendo que la oficina se encuentra en el centro de la zona de oficinas (ST_Distance, ST_Centroid) 6. Buscar clientes cuya ubicación es cercana al límite de una determinada zona peligrosa (ST_Buffer, ST_Overlaps) 7. Buscar aquellos clientes de alto riesgo que están cubiertos por una determinada oficina (Todos los pasos utilizan gseRunSpatialQueries)	Estos pasos realizan un análisis espacial utilizando las funciones y predicados espaciales en lenguaje SQL de DB2. El optimizador de consultas de DB2 aprovecha el índice espacial de las columnas espaciales para mejorar el rendimiento de las consultas siempre que es posible.
Exportar capas espaciales a archivos	Exportar la capa highRiskCustomers (gseExportShape)	El paso muestra un ejemplo de exportación de los resultados de su consulta a un archivo SHAPE. Gracias a la exportación de los resultados de la consulta a otro formato de archivo, la información puede ser utilizada por una herramienta de otro proveedor (por ejemplo, ArcExplorer para Java, Versión 3.0, de ESRI).

Parte 2. Material de consulta

Capítulo 9. Procedimientos almacenados

Este capítulo documenta los procedimientos almacenados que le permiten crear un sistema de información geográfica con Spatial Extender. Cuando habilita y utiliza Spatial Extender desde el Centro de Control, invoca estos procedimientos almacenados de forma implícita. Por ejemplo, cuando pulsa **Bien** desde una ventana de Spatial Extender, DB2 llama al procedimiento o procedimientos almacenados asociados a dicha ventana. También puede invocar los procedimientos almacenados de forma explícita en un programa de aplicación. Se recomienda incluir el archivo de cabecera, db2gse.h, en un programa de este tipo. Este archivo contiene las definiciones de macros correspondientes a las constantes que asigna a los parámetros de los procedimientos almacenados. En AIX, se almacena en el directorio $DB2INSTANCE/sqllib/include/. En Windows NT, se almacena en el directorio %DB2PATH%\include\.

Atención:
Todas las constantes de series de caracteres correspondientes a parámetros de entrada de procedimientos almacenados son sensibles a mayúsculas y minúsculas. Para ver qué parámetros necesitan estas constantes, consulte las tablas de este capítulo.

Los nombres de esquema, tabla, vista, columna o de capa que asigne a un parámetro deben especificarse en mayúsculas.

Para poder invocar un procedimiento almacenado, de forma implícita o explícita, debe estar conectado a la base de datos en la que está instalado Spatial Extender. El primer procedimiento almacenado que va a utilizar es db2gse.gse_enable_db. Habilita la base de datos para operaciones espaciales. Sólo puede utilizar los otros procedimientos almacenados cuando haya habilitado la base de datos.

Las implantaciones de procedimientos almacenados se archivan en la biblioteca db2gse del servidor de Spatial Extender.

Puede utilizar la siguiente lista para buscar los procedimientos almacenados por sus nombres o por las tareas que llevan a cabo. La primera lista presenta los nombres:

- “db2gse.gse_disable_sref” en la página 90
- “db2gse.gse_enable_autogc” en la página 91
- “db2gse.gse_enable_db” en la página 95
- “db2gse.gse_enable_idx” en la página 96
- “db2gse.gse_enable_sref” en la página 98
- “db2gse.gse_export_shape” en la página 101
- “db2gse.gse_import_sde” en la página 103
- “db2gse.gse_import_shape” en la página 105
- “db2gse.gse_register_gc” en la página 108
- “db2gse.gse_register_layer” en la página 110
- “db2gse.gse_run_gc” en la página 118
- “db2gse.gse_unregist_gc” en la página 120
- “db2gse.gse_unregist_layer” en la página 121

La siguiente lista presenta las tareas que llevan a cabo los procedimientos almacenados.

- Creación de un índice para una columna espacial (consulte “db2gse.gse_enable_idx” en la página 96).
- Creación de un sistema de referencias espaciales (consulte “db2gse.gse_enable_sref” en la página 98).
- Inhabilitación de un geocodificador para que no mantenga automáticamente las columnas espaciales sincronizadas con sus correspondientes columnas de atributo (consulte “db2gse.gse_disable_autogc” en la página 86).
- Inhabilitación del soporte de operaciones espaciales en una base de datos (consulte “db2gse.gse_disable_db” en la página 89).
- Eliminación de un sistema de referencias espaciales (consulte “db2gse.gse_disable_sref” en la página 90).
- Habilitación de una base de datos para que dé soporte a operaciones espaciales (consulte “db2gse.gse_enable_db” en la página 95).
- Habilitación de un geocodificador para que mantenga automáticamente las columnas espaciales sincronizadas con sus correspondientes columnas de atributo (consulte “db2gse.gse_enable_autogc” en la página 91).
- Exportación de una capa y de su tabla asociada a un archivo de formas (consulte “db2gse.gse_export_shape” en la página 101).
- Importación de una capa y de su tabla asociada de un archivo de transferencia ESRI_SDE (consulte “db2gse.gse_import_sde” en la página 103).
- Importación de una capa y de su tabla asociada de un archivo de formas (consulte “db2gse.gse_import_shape” en la página 105).

- Registro de un codificador que no es el codificador por omisión (consulte "db2gse.gse_register_gc" en la página 108).
- Registro de una columna espacial como una capa (consulte "db2gse.gse_register_layer" en la página 110).
- Ejecución de un geocodificador en modalidad de proceso por lotes (consulte "db2gse.gse_unregist_gc" en la página 120).
- Desregistro de un codificador que no es el codificador por omisión (consulte "db2gse.gse_unregist_layer" en la página 121).
- Desregistro de una capa (consulte "db2gse.gse_unregist_layer" en la página 121).

Para obtener información sobre las secuencias en las que puede llevar a cabo estas tareas, consulte el "Capítulo 1. Acerca de Spatial Extender" en la página 3 y el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

db2gse.gse_disable_autogc

Utilice este procedimiento almacenado para eliminar o inhabilitar temporalmente los activadores que mantienen una columna espacial sincronizada con su columna o columnas de atributo. Por ejemplo, se recomienda inhabilitar los activadores mientras geocodifica los valores en la columna o columnas de atributo en modalidad de proceso por lotes. Para obtener más información sobre este tema, consulte "Acerca de la geocodificación" en la página 53.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseDisableAutoGc en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

El ID de usuario bajo el cual se invoca este procedimiento almacenado debe tener autorización en forma de autoridad, privilegio o grupo de privilegios; en concreto:

- Autorización SYSADM o DBADM para la base de datos que contiene la tabla donde están definidos los activadores que se están eliminando o inhabilitando temporalmente.
- El privilegio CONTROL para esta tabla.
- Los privilegios ALTER, SELECT y UPDATE para esta tabla.

Parámetros de entrada

Tabla 6. Parámetros de entrada para el procedimiento almacenado db2gse.gse_disable_autogc.

Nombre	Tipo de datos	Descripción
operMode	SMALLINT	Indica si los activadores se tienen que eliminar o inhabilitar temporalmente. Los activadores eliminados no tienen ningún efecto sobre las sentencias de SQL. Los activadores inhabilitados temporalmente se pueden reconstruir sin tener que especificar de nuevo parámetros definidos anteriormente. El valor de este parámetro no puede ser nulo. **Comentario:** Para eliminar activadores, utilice la macro GSE_AUTOGC_DROP. Para inhabilitarlos temporalmente, utilice la macro GSE_AUTOGC_INVALIDATE. Para saber cuáles son los valores asociados a estas macros, consulte el archivo db2gse.h. En AIX, este archivo se almacena en el directorio $DB2INSTANCE/sqllib/include/. En Windows NT, el archivo se guarda en el directorio %DB2PATH%\include\.
layerSchema	VARCHAR (30)	Nombre del esquema al que pertenece la tabla o vista especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. **Comentario:** Si no especifica un valor para el parámetro layerSchema, éste adoptará como valor por omisión el ID de usuario bajo el que se invoca el procedimiento almacenado db2gse.gse_disable_autogc.
layerTable	VARCHAR(128)	Nombre de la tabla en la que están definidos los activadores que desea eliminar o inhabilitar temporalmente. El valor de este parámetro no puede ser nulo.
layerColumn	VARCHAR(128)	Nombre de la columna habilitada para operaciones espaciales en la que se mantiene mediante los activadores que desea eliminar o inhabilitar temporalmente. El valor de este parámetro no puede ser nulo.

Parámetros de salida

Tabla 7. Parámetros de salida para el procedimiento almacenado db2gse.gse_disable_autogc.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_disable_db

Utilice este procedimiento almacenado para eliminar recursos que permiten a Spatial Extender almacenar datos espaciales y dar soporte a operaciones realizadas sobre estos datos.

El objetivo de este procedimiento almacenado consiste en ayudarle a resolver problemas o dudas que se presentan tras habilitar la base de datos para operaciones espaciales pero *antes* de añadir a la misma datos o columnas de la tabla espacial. Por ejemplo, si, tras habilitar una base de datos para operaciones espaciales, se decide utilizar Spatial Extender para otra base de datos. Puesto que aún no ha definido ninguna columna espacial ni importado datos espaciales, puede invocar este procedimiento almacenado para eliminar todos los recursos espaciales de la primera base de datos.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseDisableDB en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

El ID de usuario bajo el que se invoca este procedimiento almacenado debe tener autorización SYSADM o DBADM sobre la base de datos de la que se deben eliminar recursos de Spatial Extender.

Parámetros de salida

Tabla 8. Parámetros de salida para el procedimiento almacenado db2gse.gse_disable_db.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_disable_sref

Utilice este procedimiento almacenado para eliminar un sistema de referencias espaciales. Cuando se procesa este procedimiento almacenado, se elimina información sobre el sistema de referencias espaciales de la vista de catálogo DB2GSE.SPATIAL_REF_SYS. Para obtener información sobre esta vista, consulte la sección "DB2GSE.SPATIAL_REF_SYS" en la página 147.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseDisableSref en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

No se necesita.

Parámetro de entrada

Tabla 9. Parámetros de entrada para el procedimiento almacenado db2gse.gse_disable_sref.

Nombre	Tipo de datos	Descripción
srId	INTEGER	Identificador numérico del sistema de referencias espaciales que se tiene que eliminar. El valor de este parámetro no puede ser nulo.

Parámetros de salida

Tabla 10. Parámetros de salida para el procedimiento almacenado db2gse.gse_disable_sref.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

Restricción

Para poder eliminar un sistema de referencias espaciales, debe cancelar el registro de las capas asociadas a ese sistema. Si esas capas se mantienen sin registrar, la petición de eliminar el sistema de referencias espaciales se rechazará.

db2gse.gse_enable_autogc

Utilice este procedimiento almacenado para:

- Crear activadores que mantendrán una columna espacial sincronizada con su columna o columnas de atributo asociadas. Cada vez que se insertan valores en la columna o columnas de atributo o se actualizan valores de las mismas, un activador llamará a un geocodificador registrado para que geocodifique los valores insertados o actualizados y coloque los datos resultantes en la columna espacial.
- Volver a activar los activadores después de que se hayan inhabilitado temporalmente.
- Establecer qué función se utilizará para geocodificar los valores insertados o actualizados.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseEnableAutoGC en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

El ID de usuario bajo el cual se invoca este procedimiento almacenado debe tener autorización en forma de autoridad, privilegio o grupo de privilegios; en concreto:

- Autorización SYSADM o DBADM sobre la base de datos que contiene la tabla en la que están definidos los activadores creados por este procedimiento almacenado.
- El privilegio CONTROL sobre esta tabla.
- Los privilegios ALTER, SELECT y UPDATE sobre esta tabla.

Parámetros de entrada

Tabla 11. Parámetros de entrada para el procedimiento almacenado db2gse.gse_enable_autogc.

Nombre	Tipo de datos	Descripción
operMode	SMALLINT	Valor que indica si los activadores que inician la geocodificación se deben crear por primera vez o se deben volver a activar tras haber sido temporalmente inhabilitados. El valor de este parámetro no puede ser nulo. **Comentario:** Para crear activadores, utilice la macro GSE_AUTOGC_CREATE. Para volverlos a activar, utilice la macro GSE_AUTOGC_RECREATE. Para saber cuáles son los valores asociados a estas macros, consulte el archivo db2gse.h. En AIX, este archivo se guarda en el directorio $DB2INSTANCE/sqllib/include/. En Windows NT, el archivo se guarda en el directorio %DB2PATH%\include\.
layerSchema	VARCHAR (30)	Nombre del esquema al que pertenece la tabla especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. **Comentario:** Si no especifica un valor para el parámetro layerSchema, éste adoptará como valor por omisión el ID de usuario bajo el que se invoca el procedimiento almacenado db2gse.gse_enable_autogc.
layerTable	VARCHAR(128)	Nombre de la tabla sobre la que deben operar los activadores creados o vueltos a activar por este procedimiento almacenado. El valor de este parámetro no puede ser nulo.
layerColumn	VARCHAR(128)	Nombre de la columna espacial que deben mantener los activadores que este procedimiento almacenado crea o vuelve a activar. El valor de este parámetro no puede ser nulo.

Tabla 11. Parámetros de entrada para el procedimiento almacenado db2gse.gse_enable_autogc. (continuación)

Nombre	Tipo de datos	Descripción
gcId	INTEGER	Identificador del geocodificador que invocarán los activadores de inserción y actualización que este procedimiento almacenado crea o vuelve a activar. El valor de este parámetro no puede ser nulo si el valor del parámetro operMode es GSE_AUTOGC_CREATE. Su valor puede ser nulo si el valor de operMode es GSE_AUTOGC_RECREATE.
precisionLevel	INTEGER	Grado en que los datos fuente deben coincidir con los datos de referencia correspondientes para que el geocodificador procese satisfactoriamente los datos fuente. El valor de este parámetro no puede ser nulo si el valor del parámetro operMode es GSE_AUTOGC_CREATE. Su valor puede ser nulo si el valor de operMode es GSE_AUTOGC_RECREATE. **Comentario:** El nivel de precisión puede estar comprendido entre el 1 y el 100 por ciento.
vendorSpecific	VARCHAR(256)	Información técnica suministrada por el proveedor; por ejemplo, la vía de acceso y el nombre de un archivo que utiliza el proveedor para definir parámetros. El valor de este parámetro no puede ser nulo si el valor del parámetro operMode es GSE_AUTOGC_CREATE. Su valor puede ser nulo si el valor de operMode es GSE_AUTOGC_RECREATE.

Parámetros de salida

Tabla 12. Parámetros de salida para el procedimiento almacenado db2gse.gse_enable_autogc.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

Restricciones

- El parámetro layerColumn debe hacer referencia a una columna registrada como una capa de tabla.
- Si el valor del parámetro operMode es GSE_AUTOGC_CREATE, debe asignar un identificador de un geocodificador registrado al parámetro gcId.

db2gse.gse_enable_db

Utilice este procedimiento almacenado para suministrar a una base de datos los recursos que necesita para almacenar datos espaciales y para dar soporte a operaciones. Estos recursos incluyen tipos de datos espaciales, un tipo de índice espacial, vistas y tablas de catálogo, funciones suministradas y otros procedimientos almacenados. La biblioteca externa y el nombre de función de este procedimiento almacenado es db2gse.gse_enable_db.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

El ID de usuario bajo el que se invoca el procedimiento almacenado debe tener autorización SYSADM o DBADM sobre la base de datos que se está habilitando.

Parámetros de salida

Tabla 13. Parámetros de salida para el procedimiento almacenado db2gse.gse_enable_db.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_enable_idx

Utilice este procedimiento almacenado para crear un índice para una columna espacial.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseEnableIdx en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

El ID de usuario bajo el que se invoca este procedimiento almacenado debe tener uno de los siguientes privilegios o autorizaciones:

- Autorización SYSADM o DBADM sobre la base de datos que contiene la tabla para la que se va a utilizar el índice habilitado.
- El privilegio CONTROL o INDEX sobre esta tabla.

Parámetros de entrada

Tabla 14. Parámetros de entrada para el procedimiento almacenado db2gse.gse_enable_idx.

Nombre	Tipo de datos	Descripción
layerSchema	VARCHAR (30)	Nombre del esquema al que pertenece la tabla especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. **Comentario:** Debe proporcionar un valor para este parámetro. El parámetro puede ser un valor nulo.
layerTable	VARCHAR(128)	Nombre de la tabla en la que se va a definir el índice que está creando. El valor de este parámetro no puede ser nulo.
layerColumn	VARCHAR(128)	Nombre de la columna habilitada para operaciones espaciales que se va a buscar con la ayuda del índice que está creando. El valor de este parámetro no puede ser nulo.
indexName	VARCHAR(128)	Nombre del índice que se va a crear. El valor de este parámetro no puede ser nulo. **Comentario:** No especifique ningún nombre de esquema. Spatial Extender asigna automáticamente el índice al esquema al que se hacer referencia en el parámetro layerSchema.

Tabla 14. Parámetros de entrada para el procedimiento almacenado db2gse.gse_enable_idx. (continuación)

Nombre	Tipo de datos	Descripción
gridSize1	DOUBLE	Número que indica la granularidad que debe tener la cuadrícula de índice más fina. El valor de este parámetro no puede ser nulo.
gridSize2	DOUBLE	Número que indica (1) que no debe haber una segunda cuadrícula para este índice o (2) cuál debe ser la granularidad de la segunda cuadrícula. El valor de este parámetro puede ser nulo. **Comentario:** Si no debe haber una segunda cuadrícula de índice, especifique 0. Si desea una segunda cuadrícula, su granularidad debe ser menor que la indicada por gridSize1.
gridSize3	DOUBLE	Número que indica (1) que no debe haber una tercera cuadrícula para este índice o (2) cuál debe ser la granularidad de la tercera cuadrícula. El valor de este parámetro puede ser nulo. **Comentario:** Si no debe haber una tercera cuadrícula de índice, especifique 0. Si desea una tercera cuadrícula, su granularidad debe ser menor que la indicada por gridSize2.

Parámetros de salida

Tabla 15. Parámetros de salida para el procedimiento almacenado db2gse.gse_enable_idx.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_enable_sref

Utilice este procedimiento almacenado para especificar cómo se deben convertir los números negativos y decimales de un sistema de coordenadas especificado en enteros positivos para que Spatial Extender los pueda almacenar. Sus especificaciones se denominan en conjunto *sistema de referencias espaciales*. Cuando se procesa este procedimiento almacenado, se añade información sobre el sistema de referencias espaciales a la vista de catálogo DB2GSE.SPATIAL_REF_SYS. Para obtener información sobre esta vista, consulte la sección "DB2GSE.SPATIAL_REF_SYS" en la página 147.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseEnableSref en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

No se necesita.

Parámetros de entrada

Tabla 16. Parámetros de entrada para el procedimiento almacenado db2gse.gse_enable_sref.

Nombre	Tipo de datos	Descripción
srId	INTEGER	Un identificador numérico para el sistema de referencias espaciales. El valor de este parámetro no puede ser nulo. **Comentario:** Este identificador debe ser exclusivo dentro de su base de datos habilitada para operaciones espaciales.
srName	VARCHAR(64)	Breve descripción del sistema de referencias espaciales. El valor de este parámetro no puede ser nulo. **Comentario:** Esta descripción debe ser exclusiva dentro de su base de datos habilitada para operaciones espaciales.
falsex	DOUBLE	Un número que, al restarse de un valor negativo de coordenada X, da como resultado un número no negativo (es decir, un número positivo o un cero). El valor de este parámetro no puede ser nulo.

Tabla 16. Parámetros de entrada para el procedimiento almacenado db2gse.gse_enable_sref. (continuación)

Nombre	Tipo de datos	Descripción
falsey	DOUBLE	Un número que, al restarse de un valor negativo de coordenada Y, da como resultado un número no negativo (es decir, un número positivo o un cero). El valor de este parámetro no puede ser nulo.
xyunits	DOUBLE	Un número que, cuando se multiplica por una coordenada X decimal o por una coordenada Y decimal, da como resultado un entero que se puede almacenar como un elemento de datos de 32 bits. El valor de este parámetro no puede ser nulo.
falsez	DOUBLE	Un número que, al restarse de un valor negativo de coordenada Z, da como resultado un número no negativo (es decir, un número positivo o un cero). El valor de este parámetro no puede ser nulo.
zunits	DOUBLE	Un número que, cuando se multiplica por una coordenada Z decimal, da como resultado un entero que se puede almacenar como un elemento de datos de 32 bits. El valor de este parámetro no puede ser nulo.
falsem	DOUBLE	Un número que, al restarse de una medida negativa, da como resultado un número no negativo (es decir, un número positivo o un cero). El valor de este parámetro no puede ser nulo.
munits	DOUBLE	Un número que, cuando se multiplica por una medida decimal, da como resultado un entero que se puede almacenar como un elemento de datos de 32 bits. El valor de este parámetro no puede ser nulo.

Tabla 16. Parámetros de entrada para el procedimiento almacenado db2gse.gse_enable_sref. (continuación)

Nombre	Tipo de datos	Descripción
scId	INTEGER	Identificador numérico del sistema de coordenadas a partir del cual se obtiene el sistema de referencias espaciales. Para saber qué es un identificador numérico del sistema de coordenadas, consulte la vista de catálogo DB2GSE.COORD_REF_SYS en "DB2GSE.COORD_REF_SYS" en la página 145. El valor de este parámetro no puede ser nulo.

Parámetros de salida

Tabla 17. Parámetros de salida para el procedimiento almacenado db2gse.gse_enable_sref.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_export_shape

Utilice este procedimiento almacenado para exportar una capa y su tabla asociada a un archivo de formas o para crear un nuevo archivo de formas y exportar una capa y su tabla asociada a este nuevo archivo.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseExportShape en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

El ID de usuario bajo el cual se invoca este procedimiento almacenado debe tener el privilegio SELECT sobre la tabla que se va a exportar.

Parámetros de entrada

Tabla 18. Parámetros de entrada para el procedimiento almacenado db2gse.gse_export_shape.

Nombre	Tipo de datos	Descripción
layerSchema	VARCHAR (30)	Nombre del esquema al que pertenece la tabla especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. **Comentario:** Si no especifica un valor para el parámetro layerSchema, éste adoptará como valor por omisión el ID de usuario bajo el que se invoca el procedimiento almacenado db2gse.gse_export_shape.
layerTable	VARCHAR(128)	Nombre de la tabla que está exportando. El valor de este parámetro no puede ser nulo.
layerColumn	VARCHAR (30)	Nombre de la columna que se ha registrado como la capa que está exportando. El valor de este parámetro no puede ser nulo.
fileName	VARCHAR(128)	Nombre del archivo de formas al que se va a exportar la capa especificada. El valor de este parámetro no puede ser nulo.

Tabla 18. Parámetros de entrada para el procedimiento almacenado db2gse.gse_export_shape. (continuación)

Nombre	Tipo de datos	Descripción
whereClause	VARCHAR(1024)	Cuerpo de la cláusula WHERE. Define una restricción sobre el grupo de registros que se deben exportar. La cláusula puede hacer referencia a cualquier columna de atributo de la tabla que está exportando. La palabra clave WHERE no es necesaria en esta cláusula. El valor de este parámetro puede ser nulo.

Parámetros de salida

Tabla 19. Parámetros de salida para el procedimiento almacenado db2gse.gse_export_shape.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

Restricción

No puede exportar más de una capa simultáneamente.

db2gse.gse_import_sde

Utilice este procedimiento almacenado para importar un archivo de transferencia SDE a una base de datos que se ha habilitado para operaciones espaciales. El procedimiento almacenado puede funcionar de dos formas:

- Si el archivo de transferencia SDE va destinado a una tabla existente que tiene una columna de capa registrada, Spatial Extender cargará la tabla con los datos del archivo.
- Si no es así, Spatial Extender creará una tabla con una columna espacial, registrará esta columna como una capa y cargará la capa y las otras columnas de la tabla con los datos del archivo.

El sistema de referencias espaciales especificado en el archivo de transferencia SDE se comparará con los sistemas de referencias espaciales registrados ante Spatial Extender. Si el sistema especificado coincide con un sistema registrado, los valores negativos y decimales de los datos de trasferencia se modificarán, cuando se carguen, tal como indique el sistema registrado. Si el sistema especificado no coincide con ninguno de los sistemas registrados, Spatial Extender creará un nuevo sistema de referencias espaciales para indicar las modificaciones.

Autorización

Cuando importa datos a una tabla existente, el ID de usuario bajo el que se invoca este procedimiento almacenado debe tener una de las siguientes autorizaciones o privilegios:

- Autorización SYSADM o DBADM sobre la base de datos que contiene la tabla en la que se deben importar los datos.
- El privilegio CONTROL sobre esta tabla.

Cuando la tabla en la que desea importar datos se debe crear, el ID de usuario bajo el que se invoca este procedimiento almacenado debe tener una de las siguientes autorizaciones o privilegios.

- Autorización SYSADM o DBADM sobre la base de datos que contiene la tabla que se va a crear.

Parámetros de entrada

Tabla 20. Parámetros de entrada del procedimiento almacenado db2gse.gse_import_sde.

Nombre	Tipo de datos	Descripción
layerSchema	VARCHAR (30)	Nombre del esquema al que pertenece la tabla o vista especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. Deber tener 30 caracteres o menos. **Comentario:** Si no especifica un valor para el parámetro layerSchema, éste adoptará como valor por omisión el ID de usuario bajo el que se invoca el procedimiento almacenado db2gse.gse_import_sde.
layerTable	VARCHAR(128)	Nombre de la tabla en la que se deben cargar los datos de transferencia SDE. El valor de este parámetro no puede ser nulo.
layerColumn	VARCHAR (30)	Nombre de la columna que se ha registrado como la capa en la que se deben cargar los datos espaciales del archivo de transferencia SDE. El valor de este parámetro no puede ser nulo. Deber tener 30 caracteres o menos.
fileName	VARCHAR(128)	Nombre del archivo de transferencia SDE que se debe importar. El valor de este parámetro no puede ser nulo.
commitScope	INTEGER	Número de registros por punto de control. El valor de este parámetro puede ser nulo.

Parámetros de salida

Tabla 21. Parámetros de salida del procedimiento almacenado db2gse.gse_import_sde.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_import_shape

Utilice este procedimiento almacenado para importar un archivo de formas ESRI a una base de datos que se ha habilitado para operaciones espaciales. El procedimiento almacenado puede funcionar de dos formas:

- Si el archivo de formas va destinado a una tabla existente que tiene una columna de capa registrada, Spatial Extender cargará la tabla con los datos del archivo.
- Si no es así, Spatial Extender creará una tabla con una columna espacial, registrará esta columna como una capa y cargará la capa y las otras columnas de la tabla con los datos del archivo.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseImportShape en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Cuando importa un conjunto de representaciones de formas ESRI, recibe al menos dos archivos. Todos los archivos tienen el mismo nombre, pero extensiones diferentes. Por ejemplo, las extensiones de los dos archivos que recibe siempre son .shp y .shx.

Para recibir los archivos correspondientes a un conjunto de representaciones de forma, asigne al parámetro fileName el nombre que los archivos tienen en común. No especifique una extensión de archivo. De esta forma, se asegura la importación de todos los archivos que necesita: el archivo .shp, el archivo .shx y cualquier otro que pudiera incluirse.

Por ejemplo, suponga que un conjunto de representaciones de formas ESRI está almacenado en los archivos llamados Lakes.shp y Lakes.shx. Cuando importe estas representaciones, especificará sólo "Lakes" para el parámetro fileName.

Los archivos de transferencia SDE tienen nombres, pero carecen de extensión. Por tanto, cuando importe un archivo de transferencia SDE, especifique su nombre, pero sin extensión, para el parámetro fileName.

Autorización

El ID de usuario bajo el que se invoca este procedimiento almacenado debe tener uno de los siguientes privilegios o autorizaciones:

- Autorización SYSADM o DBADM sobre la base de datos que contiene la tabla en la que se van a cargar los datos de forma importados.
- El privilegio CONTROL sobre esta tabla.

Parámetros de entrada

Tabla 22. Parámetros de entrada para el procedimiento almacenado db2gse.gse_import_shape.

Nombre	Tipo de datos	Descripción
layerSchema	VARCHAR (30)	Nombre del esquema al que pertenece la tabla o vista especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. **Comentario:** Si no especifica un valor para el parámetro layerSchema, éste adoptará como valor por omisión el ID de usuario bajo el que se invoca el procedimiento almacenado db2gse.gse_import_shape.
layerTable	VARCHAR(128)	Nombre de la tabla en la que se va a cargar el archivo de formas importado. El valor de este parámetro no puede ser nulo.
layerColumn	VARCHAR (30)	Nombre de la columna que se ha registrado como la capa en la que se van a cargar los datos de forma. El valor de este parámetro no puede ser nulo.
fileName	VARCHAR(128)	Nombre del archivo de formas que se va a importar. El valor de este parámetro no puede ser nulo.
exceptionFile	VARCHAR(128)	Vía de acceso y nombre del archivo en el que se almacenan las formas que no se han podido importar. Es un archivo nuevo que se creará cuando se ejecute el procedimiento almacenado db2gse.gse_import_shape. Especifique un nombre de archivo, sin extensión, para el parámetro exceptionFile. El valor de este parámetro no puede ser nulo.
srId	INTEGER	Identificador del sistema de referencias espaciales que se debe utilizar para la capa en la que se van a cargar los datos de forma. El valor de este parámetro puede ser nulo. **Comentario:** Si no se especifica este identificador, la transformación interna se definirá con la máxima resolución posible para el archivo de formas.

Tabla 22. Parámetros de entrada para el procedimiento almacenado db2gse.gse_import_shape. (continuación)

Nombre	Tipo de datos	Descripción
commitScope	INTEGER	Número de registros por punto de control. El valor de este parámetro puede ser nulo.

Parámetros de salida

Tabla 23. Parámetros de salida del procedimiento almacenado db2gse.gse_import_shape.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_register_gc

Utilice este procedimiento almacenado para registrar un geocodificador que no sea el geocodificador por omisión. Para saber si un geocodificador ya se ha registrado, consulte la vista de catálogo DB2GSE.SPATIAL_GEOCODER (descrita en el tema "DB2GSE.SPATIAL_GEOCODER" en la página 147).

Autorización

El ID de usuario bajo el que se invoca este procedimiento almacenado debe tener autorización SYSADM o DBADM sobre la base de datos que contiene el geocodificador que registra este procedimiento almacenado.

Parámetros de entrada

Tabla 24. Parámetros de entrada para el procedimiento almacenado db2gse.gse_register_gc.

Nombre	Tipo de datos	Descripción
gcId	INTEGER	Identificador numérico del geocodificador que está registrando. El valor de este parámetro no puede ser nulo. **Comentario:** Este identificador debe ser exclusivo dentro de la base de datos.
gcName	VARCHAR(64)	Breve descripción del geocodificador que está registrando. El valor de este parámetro no puede ser nulo. **Comentario:** Esta descripción debe ser una serie de caracteres exclusiva dentro de la base de datos.
vendorName	VARCHAR(64)	Nombre del proveedor que ha suministrado el geocodificador que está registrando. El valor de este parámetro no puede ser nulo.
primaryUDF	VARCHAR(256)	Nombre calificado al completo del geocodificador que está registrando. El valor de este parámetro no puede ser nulo.

Tabla 24. Parámetros de entrada para el procedimiento almacenado db2gse.gse_register_gc. (continuación)

Nombre	Tipo de datos	Descripción
precisionLevel	INTEGER	Grado en que los datos fuente deben coincidir con los datos de referencia correspondientes para que el geocodificador procese satisfactoriamente los datos fuente. El valor de este parámetro no puede ser nulo. **Comentario:** El nivel de precisión puede estar comprendido entre 1 y 100 por ciento.
vendorSpecific	VARCHAR(256)	Información técnica suministrada por el proveedor; por ejemplo, la vía de acceso y el nombre de un archivo que utiliza el proveedor para definir parámetros. El valor de este parámetro puede ser nulo.
geoArea	VARCHAR(256)	Zona geográfica que se debe geocodificar. El valor de este parámetro puede ser nulo.
description	VARCHAR(256)	Observaciones suministradas por el proveedor. El valor de este parámetro puede ser nulo.

Parámetros de salida

Tabla 25. Parámetros de salida para el procedimiento almacenado db2gse.gse_register_gc.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_register_layer

Utilice este procedimiento almacenado para registrar una columna espacial como una capa. Cuando se procesa este procedimiento almacenado, se añade información sobre la capa que se registra a la vista de catálogo DB2GSE.GEOMETRY_COLUMNS. Para obtener información sobre esta vista, consulte la sección "DB2GSE.GEOMETRY_COLUMNS" en la página 146.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseRegisterLayer en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Este procedimiento almacenado no es efectivo para los tipos de tabla siguientes:

- A = Alias (seudónimo)
- H = Hierarchy table (tabla jerárquica)
- N = Nickname (apodo)
- S = Summary table (tabla de resumen)
- U = Typed table (tabla con tipo)
- W = Typed view (vista con tipo)

Autorización

El ID de usuario bajo el que se invoca este procedimiento almacenado debe tener uno de los siguientes privilegios o autorizaciones:

- Para una capa de tabla:
  - Autorización SYSADM o DBADM sobre la base de datos que contiene la tabla a la que pertenece esta capa.
  - El privilegio CONTROL o ALTER sobre esta tabla.
- Para una capa de vista:
  - El privilegio SELECT sobre la tabla o tablas base que contienen (1) los datos de dirección que se deben geocodificar para esta capa y (2) los datos espaciales resultantes de la geocodificación.

Parámetros de entrada

Tabla 26. Parámetros de entrada para el procedimiento almacenado db2gse.gse_register_layer.

Nombre	Tipo de datos	Descripción
layerSchema	INTEGER(30)	Nombre del esquema al que pertenece la tabla o vista especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. **Comentario:** Si no especifica un valor para el parámetro layerSchema, éste adoptará como valor por omisión el ID de usuario bajo el que se invoca el procedimiento almacenado db2gse.gse_register_layer.
layerTable	VARCHAR(128)	Nombre de la tabla o vista que contiene la columna que se registra como una capa. El valor de éste parámetro no puede ser nulo.
layerColumn	VARCHAR(128)	Nombre de la columna que se está registrando como una capa. Para una tabla, si la columna no existe, Spatial Extender la añadirá utilizando la sentencia ALTER. Para una vista, la columna debe ya existir. Se puede especificar una sola columna para el parámetro layerColumn. Por tanto, cuando registre como capa varias columnas de una tabla o vista, debe ejecutar este procedimiento almacenado para cada columna por separado. El valor de este parámetro no puede ser nulo.
layerTypeName	VARCHAR(64)	Tipo de datos de la columna que se está registrando como una capa. Sólo se aceptan los tipos de datos proporcionados por Spatial Extender. Debe especificar el tipo de datos en mayúsculas; por ejemplo: `ST_POINT` No es necesario que especifique un nombre de esquema, pues éste se añade automáticamente. El valor de este parámetro no puede ser nulo si la columna es una columna de tabla que se debe crear cuando se procese este procedimiento almacenado. En otro caso, si la columna es una columna existente dentro de una tabla o vista, el valor de este parámetro puede ser nulo.

Tabla 26. Parámetros de entrada para el procedimiento almacenado db2gse.gse_register_layer. (continuación)

Nombre	Tipo de datos	Descripción
srId	INTEGER	Identificador del sistema de referencias espaciales utilizado para esta capa. El valor de este parámetro no puede ser nulo para una capa de tabla. Spatial Extender pasa por alto este parámetro cuando el usuario registra una capa de vista.
geoSchema	VARCHAR (30)	Se aplica cuando se registra como capa una columna de vista. El parámetro geoSchema es el esquema de la tabla asociado a la vista a la que pertenece la columna. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de vista. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de tabla. Las vistas basadas en más de una tabla base o en otras vistas no pueden utilizarse con este parámetro. **Comentario:** Si no especifica un valor para el parámetro geoSchema, éste adoptará como valor por omisión el valor del parámetro layerSchema.
geoTable	VARCHAR(128)	Se aplica cuando el usuario registra una columna de vista como una capa. El parámetro geoTable es el nombre de la tabla asociada a la vista a la que pertenece la columna. Las vistas basadas en más de una tabla base o en otras vistas no pueden utilizarse con este parámetro. El valor de este parámetro no puede ser nulo cuando registra una columna de vista como una capa. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de tabla.

Tabla 26. Parámetros de entrada para el procedimiento almacenado db2gse.gse_register_layer. (continuación)

Nombre	Tipo de datos	Descripción
geoColumn	VARCHAR(128)	Se aplica cuando se registra como capa una columna de vista. El parámetro geoColumn es el nombre de la columna de tabla asociada a esta columna de vista. Las vistas basadas en más de una tabla base o en otras vistas no pueden utilizarse con este parámetro. El valor de este parámetro no puede ser nulo cuando registra una columna de vista como una capa. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de tabla.
nAttributes	SMALLINT	Número de columnas que contienen los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista.
attr1Name	VARCHAR(128)	Nombre de la primera columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. Si desea poder utilizar el geocodificador por omisión, debe guardar las direcciones de calles en la columna attr1Name.

Tabla 26. Parámetros de entrada para el procedimiento almacenado db2gse.gse_register_layer. (continuación)

Nombre	Tipo de datos	Descripción
attr2Name	VARCHAR(128)	Nombre de la segunda columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. Si desea poder utilizar el geocodificador por omisión, debe guardar los nombres de ciudades en la columna attr2Name.
attr3Name	VARCHAR(128)	Nombre de la tercera columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. Si desea poder utilizar el geocodificador por omisión, debe guardar los nombres o abreviaturas de los estados (provincias) en la columna attr3Name.
attr4Name	VARCHAR(128)	Nombre de la cuarta columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. Si desea poder utilizar el geocodificador por omisión, debe guardar los códigos postales en la columna attr4Name.

Tabla 26. Parámetros de entrada para el procedimiento almacenado db2gse.gse_register_layer. (continuación)

Nombre	Tipo de datos	Descripción
attr5Name	VARCHAR(128)	Nombre de la quinta columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. El geocodificador por omisión pasa por alto la columna Attr5Name.
attr6Name	VARCHAR(128)	Nombre de la sexta columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. El geocodificador por omisión pasa por alto la columna Attr6Name.
attr7Name	VARCHAR(128)	Nombre de la séptima columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. El geocodificador por omisión pasa por alto la columna Attr7Name.

Tabla 26. Parámetros de entrada para el procedimiento almacenado db2gse.gse_register_layer. (continuación)

Nombre	Tipo de datos	Descripción
attr8Name	VARCHAR(128)	Nombre de la octava columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. El geocodificador por omisión pasa por alto la columna Attr8Name.
attr9Name	VARCHAR(128)	Nombre de la novena columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. El geocodificador por omisión pasa por alto la columna Attr9Name.
attr10Name	VARCHAR(128)	Nombre de la décima columna que contiene los datos fuente que se deben geocodificar para esta capa. El valor de este parámetro puede ser nulo cuando se registra como capa una columna de tabla. Spatial Extender pasa por alto este parámetro cuando se registra como capa una columna de vista. El geocodificador por omisión pasa por alto la columna Attr10Name.

Parámetros de salida

Tabla 27. Parámetros de salida del procedimiento almacenado db2gse.gse_register_layer.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

Restricciones

- Si está registrando una columna de vista como una capa, ésta se debe basar en una columna de tabla que ya se haya registrado como una capa.
- No puede haber más de diez columnas de atributo que contengan los datos que se deben geocodificar para la capa que está registrando.

db2gse.gse_run_gc

Utilice este procedimiento almacenado para ejecutar un geocodificador en modalidad de proceso por lotes. Para obtener información sobre esta tarea, consulte la sección "Ejecución del geocodificador en la modalidad de proceso por lotes" en la página 57.

Para ver un ejemplo del código que sirve para invocar este procedimiento almacenado, consulte la función C gseRunGC en el programa de ejemplo. Para obtener información sobre este programa, consulte el "Capítulo 8. Escritura de aplicaciones para Spatial Extender" en la página 73.

Autorización

El ID de usuario bajo el que se invoca este procedimiento almacenado debe tener uno de los siguientes privilegios o autorizaciones:

- Autorización SYSADM o DBADM sobre la base de datos que contiene la tabla sobre la que va a operar el geocodificador especificado.
- El privilegio CONTROL o UPDATE sobre esta tabla.

Parámetros de entrada

Tabla 28. Parámetros de entrada para el procedimiento almacenado db2gse.gse_run_gc.

Nombre	Tipo de datos	Descripción
layerSchema	VARCHAR (30)	Nombre del esquema al que pertenece la tabla o vista especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. **Comentario:** Si no especifica un valor para el parámetro layerSchema, éste adoptará como valor por omisión el ID de usuario bajo el que se invoca el procedimiento almacenado db2gse.gse_run_gc.
layerTable	VARCHAR(128)	Nombre de la tabla que contiene la columna en la que se van a insertar los datos geocodificados. El valor de este parámetro no puede ser nulo.
layerColumn	VARCHAR(128)	Nombre de la columna en la que se van a insertar los datos geocodificados. El valor de este parámetro no puede ser nulo.

Tabla 28. Parámetros de entrada para el procedimiento almacenado db2gse.gse_run_gc. (continuación)

Nombre	Tipo de datos	Descripción
gcId	INTEGER	Identificador del geocodificador que desea ejecutar. El valor de este parámetro puede ser nulo. Para averiguar los identificadores de los geocodificadores registrados, consulte la vista de catálogo DB2GSE.SPATIAL_GEOCODER.
precisionLevel	INTEGER	Grado en que los datos fuente deben coincidir con los datos de referencia correspondientes para que el geocodificador procese satisfactoriamente los datos fuente. El valor de este parámetro puede ser nulo. **Comentario:** El nivel de precisión puede estar comprendido entre 1 y 100 por ciento.
vendorSpecific	VARCHAR(256)	Información técnica suministrada por el proveedor; por ejemplo, la vía de acceso y el nombre de un archivo que utiliza el proveedor para definir parámetros. El valor de este parámetro puede ser nulo.
whereClause	VARCHAR(256)	La parte central de la cláusula WHERE. Define una restricción sobre el grupo de registros que se deben geocodificar. La cláusula puede hacer referencia a cualquier columna de atributo de la tabla sobre la que opera el geocodificador. El valor de este parámetro puede ser nulo.
commitScope	INTEGER	Número de registros por punto de control. El valor de este parámetro puede ser nulo.

Parámetros de salida

Tabla 29. Parámetros de salida del procedimiento almacenado db2gse.gse_run_gc.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_unregist_gc

Utilice este procedimiento almacenado para desregistrar un geocodificador que no sea el geocodificador por omisión.

Para buscar información sobre el geocodificador que desea desregistrar, consulte la vista de catálogo DB2GSE.SPATIAL_GEOCODER, consulte la sección "DB2GSE.SPATIAL_GEOCODER" en la página 147.

Autorización

El ID de usuario bajo el que se invoca este procedimiento almacenado debe tener autorización SYSADM o DBADM sobre la base de datos que contiene el geocodificador que se va a desregistrar.

Parámetro de entrada

Tabla 30. Parámetros de entrada para el procedimiento almacenado db2gse.gse_unregist_gc.

Nombre	Tipo de datos	Descripción
gcId	INTEGER	El identificador del geocodificador que se va a desregistrar. El valor de este parámetro no puede ser nulo.

Parámetros de salida

Tabla 31. Parámetros de salida para el procedimiento almacenado db2gse.gse_unregist_gc.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

db2gse.gse_unregist_layer

Utilice este procedimiento almacenado para desregistrar una capa. Para ello, el procedimiento almacenado:

- Elimina la definición de la capa de las tablas de catálogo de Spatial Extender.
- Suprime la restricción de control que Spatial Extender ha colocado en la tabla base de esta capa para asegurar que los datos espaciales de la capa cumplen con los requisitos del sistema de referencias espaciales de la capa.
- Elimina los activadores que se utilizan para actualizar la columna espacial cuando se añaden, modifican o eliminan datos de dirección.

Cuando se geocodifican los datos de dirección contenidos en una fila de tabla, los datos espaciales resultantes se colocan en la misma fila. Por tanto, si se suprime la fila, también se suprimen los datos de dirección y los datos espaciales. Los activadores no suprimen datos espaciales.

Cuando se procesa el procedimiento almacenado, se elimina la información referente a la capa en la vista de catálogo DB2GSE.GEOMETRY_COLUMNS. Para obtener información sobre esta vista, consulte la sección "DB2GSE.GEOMETRY_COLUMNS" en la página 146.

Autorización

El ID de usuario bajo el que se invoca este procedimiento almacenado debe tener uno de los siguientes privilegios o autorizaciones:

- Para una capa de tabla:
  - Autorización SYSADM o DBADM sobre la base de datos que contiene la tabla base de esta capa.
  - El privilegio CONTROL o ALTER sobre esta tabla.
- Para una capa de vista:
  - El privilegio SELECT sobre la tabla o tablas base que contienen (1) los datos de direcciones que se geocodifican para esta capa y (2) los datos espaciales resultantes de la geocodificación.

Parámetros de entrada

Tabla 32. Parámetros de entrada para el procedimiento almacenado db2gse.gse_unregist_layer.

Nombre	Tipo de datos	Descripción
layerSchema	VARCHAR (30)	Nombre del esquema al que pertenece la tabla especificada en el parámetro layerTable. El valor de este parámetro puede ser nulo. **Comentario:** Si no especifica un valor para el parámetro layerSchema, éste adoptará como valor por omisión el ID de usuario bajo el que se invoca el procedimiento almacenado db2gse.gse_unregister_layer. Los nombres de esquema, tabla, vista, columna o de capa que asigne a un parámetro deben especificarse en mayúsculas.
layerTable	VARCHAR(128)	Nombre de la tabla que contiene la columna especificada en el parámetro layerColumn. El valor de este parámetro no puede ser nulo.
layerColumn	VARCHAR(128)	Nombre de la columna espacial que se ha definido como la capa que desea desregistrar. El valor de este parámetro no puede ser nulo. **Comentario:** Se puede especificar una sola capa para el parámetro layerColumn. Por tanto, cuando desregistre el registro de varias capas en una tabla o vista, debe ejecutar este procedimiento almacenado para cada capa por separado.

Parámetros de salida

Tabla 33. Parámetros de salida para el procedimiento almacenado db2gse.gse_unregist_layer.

Nombre	Tipo de datos	Descripción
msgCode	INTEGER	Código asociado a los mensajes que puede devolver el emisor de la llamada a este procedimiento almacenado.
msgText	VARCHAR(1024)	Mensaje de error completo, tal como está definido en el servidor de Spatial Extender.

Restricción

Si una columna de vista que está definida como capa de vista se basa en una columna de tabla definida como capa de tabla, no puede desregistrar esta capa de tabla hasta que desregistre la capa de vista.

Capítulo 10. Mensajes

DB2 Spatial Extender genera mensajes procedentes de:

- El centro de Control
- Procedimientos almacenados
- Funciones espaciales

Cada mensaje tiene un identificador de mensaje que consta de un prefijo y un número de mensaje. El número de mensaje también se denomina *SQLCODE*.

Existen tres tipos de mensaje: de error, de aviso e informativos. Los identificadores de mensaje que terminan con una *E* son mensajes de error. Los identificadores que terminan con una *W* indican mensajes de aviso (″warning″). Los identificadores de mensaje que terminan con una *I* son mensajes informativos.

Mensajes devueltos por el Centro de Control

El Centro de Control devuelve los mensajes siguientes. El SQLCODE de cada mensaje comienza con las letras ″DBA″.

DBA7200E Se han seleccionado más de 10 columnas como entrada de un geocodificador.

Explicación: Se pueden seleccionar un máximo de 10 columnas como entrada de un geocodificador.

Respuesta del usuario: Mueva nombres de columnas del recuadro Columnas seleccionadas al recuadro Columnas disponibles hasta que el recuadro Columnas seleccionadas contenga diez nombres o menos.

DBA7201E La base de datos no está habilitada para el funcionamiento de Spatial Extender.

Explicación: La base de datos debe estar habilitada para Spatial Extender para que pueda utilizar Spatial Extender.

Respuesta del usuario: Pulse con el botón derecho del ratón la base de datos y seleccione Spatial Extender —> Habilitar en los menús.

Mensajes devueltos por procedimientos almacenados

Los procedimientos almacenados devuelven los mensajes siguientes. El SQLCODE de cada mensaje comienza con las letras ″GSE″, seguidas por números comprendidos dentro del rango 0000-2035.

Atención: Dos de los mensajes de error (GSE2022E y GSE2035E) no están incluidos en el catálogo de mensajes de DB2 Spatial Extender. Por tanto, cuando se producen los errores correspondientes a estos mensajes, se notifica al usuario que no se pueden recuperar dichos mensajes, mediante el mensaje siguiente: SQL10007N Mensaje ″No se pudo recuperar <SQLCODE –2022 o –2035>″. Código de razón: ″4″.

GSE0000I La operación ha finalizado satisfactoriamente.

GSE0001E Spatial Extender no ha podido realizar la operación solicitada (“<nombre-operación>”) bajo el ID de usuario “<id-usuario>”.

Explicación: Ha solicitado esta operación bajo un ID de usuario que no tiene el privilegio o la autorización necesarios para llevar a cabo la operación.

Respuesta del usuario: Consulte la documentación para saber qué autorización se necesita u obténgala de un administrador de Spatial Extender.

GSE0002E “<valor>” no es un valor válido para el argumento “<nombre-argumento>”.

Explicación: El valor que ha especificado es incorrecto o está mal escrito.

Respuesta del usuario: Consulte la documentación o consulte con un administrador de Spatial Extender para saber qué valor o rango de valores tiene que especificar.

GSE0003E Spatial Extender no ha podido llevar a cabo la operación solicitada porque no se ha especificado el argumento “<nombre-argumento>”.

Explicación: No ha especificado un argumento necesario para esta operación.

Respuesta del usuario: Especifique el argumento “<nombre-argumento>” con el valor que desee; luego vuelva a solicitar la operación.

GSE0004W El argumento “<nombre-argumento>” no se ha evaluado.

Explicación: La operación que ha solicitado no utiliza el argumento “<nombre-argumento>”.

Respuesta del usuario: No se necesita.

GSE0005E Spatial Extender no ha podido procesar su solicitud de crear un objeto denominado “<nombre-objeto>”.

Explicación: El objeto “<nombre-objeto>” ya existe o no tiene el permiso adecuado para crearlo. Puede tratarse de una tabla, de una columna, de un activador, de un índice o de otro tipo de objeto.

Respuesta del usuario: Si “<nombre-objeto>” es el objeto que desea, no haga nada. Si no es así, especifique el nombre correctamente y compruebe que tiene el permiso adecuado para crear el objeto.

GSE0006E Spatial Extender no ha podido llevar a cabo la operación solicitada sobre un objeto habilitado o registrado denominado “<nombre-objeto>”.

Explicación: El objeto “<nombre-objeto>” ya está habilitado o registrado, o bien ya existe. Puede tratarse de una capa, de un índice, de un sistema de referencias espaciales, de un sistema de coordenadas, de un geocodificador o de otro tipo de objeto.

Respuesta del usuario: Asegúrese de que el objeto “<nombre-objeto>” existe y vuelva a someter su solicitud.

GSE0007E Spatial Extender no ha podido llevar a cabo la operación solicitada sobre "<nombre-objeto>", un objeto que aún no se ha habilitado o registrado.

Explicación: El objeto "<nombre-objeto>" no se ha habilitado o registrado. Puede tratarse de una capa, de un índice, de un sistema de referencias espaciales, de un sistema de coordenadas espaciales, de un geocodificador o de otro tipo de objeto.

Respuesta del usuario: Habilite o registre el objeto "<nombre-objeto>". Luego vuelva a someter su solicitud.

GSE0008E Se ha producido un error inesperado de SQL ("<mensaje-error-sql>").

Respuesta del usuario: Consulte el mensaje detallado asociado a SQLCODE en el mensaje de error de SQL "<mensaje-error-sql>". Si es necesario, póngase en contacto con el representante de servicio de IBM.

GSE0009E No se ha podido llevar a cabo la operación solicitada sobre un objeto denominado "<nombre-objeto>" que ya existe.

Explicación: "<nombre-objeto>" ya existe en la base de datos o en el sistema operativo. Puede tratarse de un archivo, de una tabla, de una vista, de una columna, de un índice, de un activador o de otro tipo de objeto.

Respuesta del usuario: Asegúrese de especificar el objeto correctamente cuando intente acceder al mismo. Si es necesario, suprima el objeto.

GSE0010E No se ha podido llevar a cabo la operación solicitada sobre un objeto denominado "<nombre-objeto>" que es posible que no exista.

Explicación: "<nombre-objeto>" no existe en la base de datos o en el sistema operativo. Puede tratarse de un archivo, de una tabla, de una vista, de una columna, de un índice, de un activador, de un archivo o de otro tipo de objeto.

Respuesta del usuario: Asegúrese de que tiene el permiso adecuado para acceder al objeto. Si tiene este permiso y el objeto no existe, tendrá que crearlo.

GSE0011E Spatial Extender no ha podido inhabilitar o desregistrar el objeto "<nombre-objeto>".

Explicación: "<nombre-objeto>" depende de otro objeto. "<nombre-objeto>" puede ser un sistema de referencias espaciales, una capa, un geocodificador u otro tipo de objeto.

Respuesta del usuario: Consulte la documentación para ver de qué tipos de objetos puede depender "<nombre-objeto>". Luego elimine el objeto específico del que depende "<nombre-objeto>".

GSE0012E Spatial Extender no ha podido procesar su solicitud porque la columna espacial calificada al completo "<esquema-capa.nombre-capa.columna-capa>" no está registrada como una capa de tabla.

Explicación: La columna espacial calificada al completo "<esquema-capa.nombre-capa.columna-capa>" debe estar registrada como una capa de tabla para que pueda realizar ciertas operaciones asociadas a la misma (por ejemplo, habilitar su índice, habilitar un geocodificador para llenarla en modalidad de proceso por lotes o para actualizarla automáticamente).

Respuesta del usuario: Asegúrese de que la columna espacial calificada al completo "<esquema-capa.nombre-capa.columna-capa>" está registrada como una capa de tabla comprobando la vista DB2GSE.GEOMETRY_COLUMNS en el catálogo de Spatial Extender. Asegúrese también de que la tabla que contiene esta columna incluye también columnas de atributo correspondientes válidas.

GSE0013E La base de datos no está habilitada para operaciones espaciales.

Explicación: La base de datos no está habilitada para operaciones espaciales. Por lo tanto, el catálogo de Spatial Extender no existe.

Respuesta del usuario: Habilite la base de datos para operaciones espaciales.

GSE0014E La base de datos ya se ha habilitado para operaciones espaciales.

Explicación: La base de datos ya se ha habilitado para operaciones espaciales.

Respuesta del usuario: Compruebe que la base de datos se ha habilitado del modo que esperaba. Si es necesario, inhabilite la base de datos.

GSE0498E Se ha producido el siguiente error: "<mensaje-error>".

GSE0499W Spatial Extender ha emitido el siguiente aviso: "<mensaje-aviso>".

GSE0500E La modalidad de operación que ha especificado ("<modalidad-operación>") no es válida.

Explicación: La modalidad especificada no recibe soporte de la operación que ha solicitado.

Respuesta del usuario: Consulte la documentación para ver qué modalidades reciben soporte de la operación.

GSE1001E Spatial Extender no ha podido registrar una capa de vista denominada "<nombre-esquema.nombre-vista.nombre-columna>" que se basa en la columna espacial "<nombre-esquema.nombre-tabla.nombre-columna>".

Explicación: La columna espacial que ha especificado ("<nombre-esquema.nombre-tabla.nombre-columna>") no se ha registrado como una capa de tabla.

Respuesta del usuario: Registre la columna "<nombre-esquema.nombre-tabla.nombre-columna>" como una capa de tabla.

GSE1002E Spatial Extender no ha podido registrar una capa de vista denominada "<nombre-esquema.nombre-vista.nombre-columna>" que se basa en la tabla "<nombre-esquema.nombre.tabla>".

Explicación: La tabla que ha especificado ("<nombre-esquema.nombre-tabla>") no es subyacente a la vista "<nombre-esquema.nombre-vista.nombre-columna>" ni directa ni indirectamente.

Respuesta del usuario: Busque cuál es la tabla base para la vista "<nombre-esquema.nombre-vista.nombre-columna>" y especifique dicha tabla.

GSE1003E Spatial Extender no ha podido acceder a la columna denominada "<nombre-columna>" de una tabla o vista denominada "<nombre-esquema.nombre-objeto>".

Explicación: La tabla o vista "<nombre-esquema.nombre-objeto>" no tiene ninguna columna denominada "<nombre-columna>".

Respuesta del usuario: Compruebe la definición de la tabla o vista "<nombre-esquema.nombre-

objeto>" para ver el nombre adecuado de la columna que desea.

GSE1004E Spatial Extender no ha podido registrar la columna espacial calificada al completo "<nombre-esquema.nombre-tabla.nombre-columna>" como una capa de tabla.

Explicación: La columna "<nombre-esquema.nombre-tabla.nombre-columna>" no tiene un tipo de datos espaciales o no está asociada a una tabla base.

Respuesta del usuario: Defina un tipo de datos espaciales para la columna "<nombre-esquema.nombre-tabla.nombre-columna>", o asegúrese de que esta columna forma parte de una tabla base local.

GSE1005E El sistema de referencias espaciales ("<id-referencias-espaciales-capa-vista>") que ha especificado para una capa de vista difiere del sistema de referencias espaciales ("<id-referencias-espaciales-capa-tabla>") que se utiliza para la capa de tabla subyacente de esta capa.

Explicación: El sistema de referencias espaciales de una capa de vista debe coincidir con el sistema de referencias espaciales de la capa de tabla subyacente.

Respuesta del usuario: Especifique el sistema de referencias espaciales de la capa de tabla subyacente para la capa de vista.

GSE1006E Puesto que "<id-referencias-espaciales>" es un ID de sistema de referencias espaciales no válido, Spatial Extender no ha podido registrar la capa que ha solicitado.

Explicación: El sistema de referencias espaciales que ha especificado ("<id-referencias-espaciales>") no se ha habilitado o registrado.

Respuesta del usuario: Habilite o registre el sistema de referencias espaciales. Luego vuelva a someter la solicitud para registrar la capa.

GSE1007E Es posible que se haya producido un error de SQL (SQLSTATE "<estadosql>") cuando Spatial Extender ha intentado, sin éxito, añadir una columna espacial ("<nombre-columna>") a la tabla "<nombre-esquema.nombre-tabla>".

Respuesta del usuario: Busque el mensaje asociado a SQLSTATE "<estadosql>".

GSE1008E Spatial Extender no ha podido registrar una capa de vista "<esquema-capa.nombre-capa.columna-capa>" porque el tipo de datos espaciales "<tipo-columna-capa>" de la capa de vista no coincide con el tipo de datos espaciales "<tipo-columna-geo>" de la capa de tabla subyacente "<esquema-geo.nombre-geo.columna-geo>".

Explicación: El tipo de datos espaciales de una capa de vista "<esquema-capa.nombre-capa.columna-capa>" debe coincidir con el tipo de datos espaciales de la capa de tabla subyacente de la capa "<esquema-geo.nombre-geo.columna-geo>". La incoherencia entre estos dos tipos de datos causa ambigüedad cuando se procesan datos espaciales.

Respuesta del usuario: Asegúrese de que los tipos de datos espaciales de la capa de vista y de su capa de tabla subyacente coinciden.

GSE1020E "<id-referencias-espaciales>" es un ID de sistema de referencias espaciales no válido.

Explicación: No se ha habilitado un sistema de referencias espaciales con un identificador "<id-referencias-espaciales>".

Respuesta del usuario: Asegúrese de que la

referencia espacial especificada se ha habilitado.

GSE1021E **Spatial Extender no ha podido habilitar el sistema de referencias espaciales "<id-referencias-espaciales>" porque el ID del sistema de coordenadas correspondiente "<id-coordenadas-espaciales>" no es válido.**

Explicación: No se ha definido ningún sistema de coordenadas con un identificador "<id-coordenadas-espaciales>" en el catálogo de Spatial Extender.

Respuesta del usuario: Compruebe el identificador del sistema de coordenadas "<id-coordenadas-espaciales>" consultando la vista DB2GSE.COORD_REF_SYS en el catálogo de Spatial Extender.

GSE1030E **Puesto que "<nombre-esquema.nombre-tabla>" no es una tabla base, Spatial Extender no ha podido habilitar un geocodificador para la misma.**

Explicación: El objeto que contiene los datos fuente que desea geocodificar debe ser una tabla base.

Respuesta del usuario: Asegúrese de que las columnas que contienen los datos fuente que desea codificar forman parte de una tabla base.

GSE1031E **Spatial Extender no ha podido habilitar el geocodificador "<id-geocodificador>" para que funcione automáticamente en modalidad de creación para la capa "<esquema-capa.nombre-capa.columna-capa>".**

Explicación: Las explicaciones posibles son las siguientes:

- El geocodificador ya está habilitado para actualizar la capa "<esquema-capa.nombre-capa.columna-capa>" automáticamente.
- El geocodificador se ha invalidado temporalmente para esta capa.
- No se ha definido ninguna columna para datos fuente para esta capa.

Respuesta del usuario: Si el geocodificador se ha invalidado temporalmente, habilítelo para que funcione automáticamente en modalidad de ″Recreación″.

GSE1032E **Spatial Extender no ha podido habilitar el geocodificador "<id-geocodificador>" para que funcione automáticamente en modalidad de recreación para la capa "<esquema-capa.nombre-capa.columna-capa>".**

Explicación: Las explicaciones posibles son las siguientes:

- El geocodificador ya está habilitado para actualizar la capa "<esquema-capa.nombre-capa.columna-capa>" automáticamente.
- El geocodificador no se ha invalidado previamente para esta capa.
- No se ha definido ninguna columna para datos fuente para esta capa.

Respuesta del usuario: Si el geocodificador se ha inhabilitado previamente en modalidad de eliminación, o si nunca se ha definido para esta capa, habilítelo para que funcione automáticamente en modalidad de ″Creación″.

GSE1033E **Se ha producido un error de SQL cuando Spatial Extender intentaba añadir activadores a una tabla que contiene la columna para la capa "<esquema-capa.nombre-capa.columna-capa>" (SQLSTATE "<estadosql>").**

Explicación: La finalidad de los activadores es mantener la integridad de los datos entre las columnas de atributo de las que procede la entrada del geocodificador y la columna espacial en la que va su salida. El error de SQL se ha producido cuando DB2 ha intentado crear estos activadores.

Respuesta del usuario: Busque el mensaje asociado a SQLSTATE "<estadosql>".

GSE1034E Spatial Extender no ha podido inhabilitar el geocodificador "<id-geocodificador>" en modalidad de eliminación para la capa "<esquema-capa.nombre-capa.columna-capa>".

Explicación: Las explicaciones posibles son las siguientes:

- El geocodificador nunca se ha habilitado para actualizar la capa "<esquema-capa.nombre-capa.columna-capa>" automáticamente.
- El geocodificador se ha inhabilitado en modalidad de eliminación.

Respuesta del usuario: Determine el estado en que estaba el geocodificador antes de que intentara inhabilitarlo. Por ejemplo, ¿estaba registrado?, ¿estaba habilitado? Luego decida si se tiene que inhabilitar en modalidad de eliminación. Por ejemplo, si nunca se ha habilitado, no hay ninguna necesidad de inhabilitarlo.

GSE1035E Spatial Extender no ha podido inhabilitar el geocodificador "<id-geocodificador>" en modalidad de invalidación para la capa "<esquema-capa.nombre-capa.columna-capa>".

Explicación: Las explicaciones posibles son las siguientes:

- El geocodificador nunca se ha habilitado para actualizar la capa "<esquema-capa.nombre-capa.columna-capa>" automáticamente.
- El geocodificador se ha inhabilitado en modalidad de invalidación o en modalidad de eliminación.

Respuesta del usuario: Determine el estado en que estaba el geocodificador antes de que intentara inhabilitarlo. Por ejemplo, ¿estaba registrado?, ¿estaba habilitado? Luego decida si se tiene que inhabilitar en modalidad de invalidación. Por ejemplo, si ya se había inhabilitado en modalidad de invalidación, no hay ninguna necesidad de inhabilitarlo por segunda vez en esta modalidad.

GSE1036E Se ha producido un error de SQL cuando Spatial Extender intentaba eliminar activadores desde una tabla que contiene la columna para la capa "<esquema-capa.nombre-capa.columna-capa>" (SQLSTATE "<estadosql>").

Explicación: Los activadores se crearon para mantener la integridad de los datos entre las columnas de atributo de las que procede la entrada del geocodificador y la columna espacial en la que va su salida. El error de SQL se ha producido cuando DB2 intentaba eliminar estos activadores.

Respuesta del usuario: Busque el mensaje asociado a SQLSTATE "<estadosql>".

GSE1037E Spatial Extender no ha podido geocodificar datos fuente para la capa de tabla "<esquema-capa.nombre-capa.columna-capa>", posiblemente porque se ha asignado un valor incorrecto "<número-de-atributos>" al argumento que especifica el número de columnas de atributo que deben suministrar datos fuente para esta capa.

Explicación: El número de columnas de atributo asociado a esta capa se ha especificado incorrectamente, o bien el nombre de una o más de estas columnas se ha especificado incorrectamente.

Respuesta del usuario: Asegúrese de que esta capa está registrada con el número y nombres correctos de columnas de atributo asociadas o bien compruebe que los datos de entrada y salida del geocodificador son correctos.

GSE1038E Se ha producido un error de SQL cuando Spatial Extender intentaba geocodificar datos fuente para la capa de tabla "<esquema-capa.nombre-capa.columna-capa>" en modalidad de proceso por lotes (SQLSTATE "<estadosql>").

Respuesta del usuario:

- Busque el mensaje asociado a SQLSTATE "<estadosql>".
- Asegúrese de que el contenido y el argumento primaryUDF de esta capa se han definido correctamente.

GSE1050E El tamaño de cuadrícula que ha especificado ("<tamaño-cuadrícula>") no es válido para el primer nivel de cuadrícula.

Explicación: Ha especificado cero o un número negativo como el tamaño de cuadrícula correspondiente al primer nivel de cuadrícula.

Respuesta del usuario: Especifique un número positivo como el tamaño de cuadrícula.

GSE1051E El tamaño de cuadrícula que ha especificado ("<tamaño-cuadrícula>") no es válido para los niveles segundo y tercero de cuadrícula.

Explicación: Ha especificado un número negativo como el tamaño de cuadrícula correspondiente al segundo o tercer nivel de cuadrícula.

Respuesta del usuario: Especifique cero o un número positivo como el tamaño de cuadrícula.

GSE1052E Se ha producido un error de SQL cuando Spatial Extender intentaba crear el índice espacial "<esquema-índice.columna-índice>" para una capa de tabla "<esquema-capa.nombre-capa.columna-capa>" (SQLSTATE "<estadosql>").

Respuesta del usuario:

- Asegúrese de que el índice espacial se ha especificado correctamente y que la columna espacial no tiene ningún índice asociado.
- Busque el mensaje asociado a SQLSTATE "<estadosql>".

GSE1500I El registro fuente "<número-registro>" se ha geocodificado satisfactoriamente.

Explicación: Un registro que contiene datos de atributo se ha geocodificado satisfactoriamente.

GSE1501W El registro fuente "<número-registro>" no se ha geocodificado.

Explicación: El nivel de precisión era demasiado alto.

Respuesta del usuario: Geocodifique con un nivel de precisión más bajo.

GSE1502W No se ha encontrado el registro fuente "<número-registro>".

Respuesta del usuario: Determine si el registro existe en la base de datos.

GSE2001E Spatial Extender no ha podido realizar la operación solicitada.

Respuesta del usuario: Consulte al administrador de la base de datos.

GSE2002E Se ha producido un error en el sistema de gestión de bases de datos.

Respuesta del usuario: Consulte al administrador de la base de datos.

GSE2003E El procedimiento almacenado que se ha invocado no se puede conectar a la estación de trabajo del usuario.

Explicación: El procedimiento almacenado no puede acceder a información que sirve para identificar la estación de trabajo ante el procedimiento.

Respuesta del usuario: Consulte al administrador de la base de datos.

GSE2004E Spatial Extender no puede validar el identificador de sistema de coordenadas especificado en el archivo de transferencia SDE que se está importando.

Respuesta del usuario: Pruebe uno o más de estos métodos:

- Compruebe que el identificador de sistema de referencias espaciales especificado en el archivo de transferencia SDE apunta al identificador de sistema de coordenadas correcto.
- Determine si el identificador correcto de sistema de coordenadas aparece listado en la vista de catálogo DB2GSE.COORD_REF_SYS. Si el identificador no está en esta vista, informe de ello al administrador de la base de datos.
- Determine si el archivo de transferencia SDE está dañado. Si está dañado, intente obtener e importar una copia inalterada de él.

GSE2005E Spatial Extender no puede validar el archivo que se desea exportar.

Explicación: Puede existir una o más razones para este problema. Por ejemplo, puede que el usuario no tenga autorización para acceder al archivo. O bien, Spatial Extender puede ser incapaz de encontrar o leer el archivo, o de reconocer tipos de datos contenidos en él.

Respuesta del usuario: Asegúrese de que especifica la vía totalmente calificada del archivo. Asegúrese también de que el ID de usuario que utiliza para ejecutar el procedimiento almacenado db2gse.gse_export_shape tiene acceso de lectura y escritura para cada directorio de la vía de acceso. Verifique que el disco donde residen estos directorios está montado en el mismo nodo donde se ejecuta DB2 y que el disco utiliza el mismo punto de montaje que el especificado en la vía de acceso. Verifique también que Spatial Extender reconoce los tipos de datos contenidos en el archivo.

Si el error se repite, intente determinar si el archivo está dañado. Si está dañado, intente obtener e importar una copia inalterada del archivo.

GSE2006E Se ha producido un error de E/S correspondiente a un archivo denominado "<nombrearchivo>".

Respuesta del usuario: Compruebe que el archivo existe, que tiene el acceso adecuado al archivo y que este no está siendo utilizado por otro usuario.

GSE2007E Spatial Extender no puede validar la capa a la que se desea importar datos.

Explicación: Puede que esté especificado incorrectamente el nombre de la columna sobre la que está definida la capa o puede que ese nombre no siga convenios estándar de denominación. Similarmente, puede que esté especificado incorrectamente el nombre de la tabla a la que pertenece la columna o puede que ese nombre no siga convenios estándar de denominación.

Respuesta del usuario: Compruebe que la capa aparece listada en la vista de catálogo DB2GSE.GEOMETRY_COLUMNS, que los nombres de la columna y de la tabla a la que pertenece están especificados correctamente, y

que estos nombres siguen convenios estándar de denominación.

GSE2008E Spatial Extender ha intentado insertar un valor nulo en una capa que tiene la restricción NOT NULL.

Respuesta del usuario: Importe la columna que contiene nulos a una capa que pueda aceptar nulos o bien solicite al administrador de la base de datos que elimine la restricción NOT NULL.

GSE2012E Spatial Extender no ha podido acceder a la capa a la que se desea importar datos.

Explicación: El ID de usuario utilizado para acceder a la capa no tiene autorización para modificar la columna sobre la que está definida la capa.

Respuesta del usuario: Solicite al administrador de la base de datos que le otorgue la autorización necesaria (por ejemplo, puede necesitar el privilegio INSERT o SELECT para la tabla a la que pertenece la columna).

GSE2014E Spatial Extender no ha podido importar ni exportar datos con respecto a la capa especificada.

Explicación: Spatial Extender no ha podido localizar la capa a la que se desea importar datos o exportar datos desde ella.

Respuesta del usuario: Determine si la capa aparece listada en la vista DB2GSE.GEOMETRY_COLUMNS. Si no está listada, utilice el procedimiento almacenado db2gse.gse_register_layer, o la ventana Crear Capa, del Centro del Control, para registrar la capa. Si la capa está listada en DB2GSE.GEOMETRY_COLUMNS, notifique el problema al administrador de la base de datos.

GSE2016E Spatial Extender no ha podido importar el archivo de formas solicitado a la capa especificada.

Explicación: El tipo de datos de los datos espaciales que se desea importar es incompatible con el tipo de datos de la capa a la cual están destinados los datos espaciales.

Respuesta del usuario: Cree una nueva capa cuyo tipos de datos sea compatible con el tipo de datos de los datos espaciales que desea importar. A continuación, importe los datos a esta nueva capa. Como alternativa, importe un archivo de formas diferente, cuyo datos espaciales sean compatibles con la capa que desea llenar.

GSE2021E Spatial Extender no ha podido acceder al archivo de formas que se desea importar.

Explicación: Existen varias razones posibles para este problema. Por ejemplo, Spatial Extender puede desconocer la vía de acceso completa del archivo de formas, o puede que no reconozca el formato del archivo, o que el disco donde reside el archivo no esté montado correctamente.

Respuesta del usuario: Asegúrese de que especifica la vía completa del archivo. Si está especificada y el error se repite, compruebe que el archivo sea realmente un archivo de formas, y no ninguna otra clase de archivo que se especificó por error como archivo de formas. Si el archivo es un archivo de formas, pruebe una de las soluciones siguientes:

- Determine si el archivo está dañado. Si está dañado, intente obtener e importar una copia inalterada del archivo.
- Si está accediendo al archivo desde otra estación de trabajo, compruebe lo siguiente:
  - El disco donde reside el archivo está montado.
  - Este disco utiliza el mismo punto de montaje que el especificado en la vía de acceso del archivo.
  - El ID de usuario utilizado en la otra estación de trabajo tiene acceso de lectura al archivo.

GSE2022E El identificador de sistema de referencias espaciales especificado no existe.

Explicación: El identificador de sistema de referencias espaciales (SRID) especificado para el archivo de formas que desea importar no está listado en el catálogo de Spatial Extender.

Respuesta del usuario: Efectúe una de las acciones siguientes:

- Asigne al archivo de formas un sistema de referencias espaciales compatible cuyo SRID ya esté registrado en la vista de catálogo DB2GSE.SPATIAL_REF_SYS.
- Como alternativa, cree un nuevo sistema de referencias espaciales para el archivo de formas. Luego, verifique que el SRID de este sistema se ha registrado en el catálogo de Spatial Extender.

Este mensaje no está incluido en el catálogo de mensajes de DB2 Spatial Extender. Cuando se produce el error correspondiente a este mensaje, se notifica al usuario que no se puede recuperar el mensaje.

GSE2023E Spatial Extender no ha podido importar datos de atributo desde el archivo de formas especificado.

Explicación: La definición de una columna de atributo del archivo de formas no se puede convertir en una definición para la columna correspondiente de la tabla a la que se desean importar datos.

Respuesta del usuario: Compruebe que el tipo de datos, la longitud máxima y otras características de esta columna de atributo se pueden convertir en sus homólogos para la columna de atributo a la que se están importando datos.

GSE2026E Spatial Extender no pudo crear un archivo para contener los datos que no pudo importar.

Explicación: Cuando se importa un archivo de formas, Spatial Extender recoge en este archivo los registros que no se pudieron importar, para que se puedan corregir e importar posteriormente. En este caso, Spatial Extender no tenía información o autorización suficiente para crear un archivo para albergar los registros rechazados.

Respuesta del usuario: Especifique la vía de acceso completa del archivo que Spatial Extender debe crear para los registros rechazados. Compruebe que no exista ya un archivo con la misma vía de acceso y el mismo nombre. Asegúrese también de que el ID de usuario que utiliza para ejecutar el procedimiento almacenado db2gse.gse_import_shape tiene acceso de lectura y escritura para cada directorio de la vía de acceso. Verifique que el disco donde residen estos directorios está montado en el mismo nodo donde se ejecuta DB2 y que el disco utiliza el mismo punto de montaje que el especificado en la vía de acceso.

GSE2027E Spatial Extender no ha podido realizar la operación de importación o exportación solicitada.

Explicación: No existe suficiente memoria para realizar la operación. Puede que el archivo que se desea importar o exportar esté dañado, lo cual produce un consumo excesivo de memoria.

Respuesta del usuario: Intente de nuevo importar o exportar el archivo. Si el error se repite, intente determinar si el archivo está dañado. Si está dañado, obtenga una copia inalterada del archivo e importe o exporte esta copia. Si el problema persiste, notifíquelo al administrador de la base de datos.

GSE2030E Spatial Extender no ha podido importar datos a la columna especificada.

Explicación: La columna especificada no se ha registrado como capa.

Respuesta del usuario: Si desea importar datos de SDE, utilice el Centro de Control de DB2 o el procedimiento almacenado db2gse.gse_import_sde para registrar la columna

como capa e importar los datos. Si desea importar datos de forma, utilice el Centro de Control o el procedimiento almacenado db2gse.gse_import_shape para registrar la columna como capa e importar los datos.

GSE2031E Spatial Extender no ha podido importar datos a la capa especificada.

Explicación: La tabla en la que se definió la capa ya no existe.

Respuesta del usuario: Si desea importar datos de SDE, utilice el Centro de Control de DB2 o el procedimiento almacenado db2gse.gse_import_sde para reconstruir la tabla e importar los datos. Si desea importar datos de forma, utilice el Centro de Control o el procedimiento almacenado db2gse.gse_import_shape para reconstruir la tabla e importar los datos.

GSE2032E Spatial Extender no ha podido importar o exportar datos de atributo.

Explicación: Una explicación posible es que se ha intentado importar datos de atributo a una tabla, pero una o más de las columnas de atributo especificadas en el archivo de importación no tienen equivalentes en esta tabla. Otra explicación posible es que se ha intentado exportar datos de una tabla o vista, pero una o más de las columnas de atributo de esta tabla o vista no tienen equivalentes en el archivo al que se desea exportar los datos.

Respuesta del usuario: Si está importando datos de atributo, identifique la columna (o cada columna) del archivo de importación que no tenga una columna equivalente en la tabla a la que se debe importar el archivo. Luego, proporcione la columna (o columnas) que falta a la tabla. Como alternativa, puede cambiar el destino de la importación para que sea una capa y utilizar conjunto distinto de columnas de atributo.

Si está exportando datos de atributo desde una tabla o vista, identifique cada columna de tabla o vista que no tenga una columna equivalente en el archivo de exportación. Luego, proporcione la columna (o columnas) que falta al archivo de exportación, o bien proporcione un nuevo archivo de exportación que contenga una columna para cada columna de datos que desea exportar.

GSE2033E Spatial Extender no ha podido leer el archivo completo que se desea importar.

Explicación: Es posible que el archivo esté dañado o truncado.

Respuesta del usuario: Intente de nuevo importar el archivo. Si el error se repite, intente obtener e importar una copia inalterada del archivo.

GSE2034E Spatial Extender no ha podido importar el archivo de transferencia SDE solicitado.

Explicación: El tipo de datos de los datos espaciales que se desea importar es incompatible con el tipo de datos de la capa a la cual están destinados los datos espaciales.

Respuesta del usuario: Cree una nueva capa cuyo tipos de datos sea compatible con el tipo de datos de los datos espaciales que desea importar. A continuación, importe los datos a esta nueva capa. Como alternativa, importe un archivo de transferencia SDE diferente, cuyo datos espaciales sean compatibles con la capa que desea llenar.

GSE2035E La coordenada especificada está fuera de los límites.

Explicación: Spatial Extender ha encontrado una coordenada que es demasiado grande o demasiado pequeña para quedar dentro de los límites del sistema de coordenadas asignado a la capa. Este problema puede surgir si el sistema de coordenadas es incompatible con datos de la capa o con su sistema asociado de referencias espaciales. El problema también podría deberse a un archivo de formas o de transferencia SDE que está dañado, o a datos anómalos que se

insertaron por error en la capa.

Respuesta del usuario: Compruebe que ha especificado el identificador correcto para el sistema de referencias espaciales utilizado por la capa especificada. Si lo especificó el identificador correcto, puede que el sistema de referencias espaciales sea incompatible con los datos de la capa o con su sistema de coordenadas asociado. De acuerdo con estas posibilidades, seleccione o cree un sistema de referencias espaciales diferente para la capa. Si el problema persiste, notifíquelo al administrador de la base de datos.

Este mensaje no está incluido en el catálogo de mensajes de DB2 Spatial Extender. Cuando se produce el error correspondiente a este mensaje, se notifica al usuario que no se puede recuperar el mensaje.

Mensajes devueltos por funciones espaciales

El SQLCODE de cada mensaje devuelto por una función espacial comienza con las letras ″GSE″, seguidas por números comprendidos dentro del rango 3001-3042.

Cuando una función espacial devuelve un mensaje, también se devuelve su valor SQLSTATE asociado, pero no su SQLCODE.

GSE3001E Error desconocido del sistema.

Explicación: Se ha producido un error inesperado del sistema.

Respuesta del usuario: Corrija la sintaxis e invoque de nuevo la función. Si el problema persiste, solicite servicio técnico.

GSE3002E Se ha especificado una cadena WKT (Well-Known Text) no válida.

Explicación: Se ha entrado una cadena de texto no válida como entrada para la función invocada.

Respuesta del usuario: Corrija la cadena de texto e invoque de nuevo la función. Para conocer el formato correcto de las cadenas de texto WKT (Well-Known Text), consulte el manual ″DB2 Spatial Extender, Guía y consulta del usuario″.

GSE3003E SRID no válido.

Explicación: El identificador de sistema de referencias espaciales (SRID) que se ha intentado pasar a la función no está listado en el catálogo del sistema de DB2 Spatial Extender.

Respuesta del usuario: Especifique un SRID que esté registrado actualmente en la vista de catálogo DB2GSE.SPATIAL_REF_SYS, o cree un sistema de referencias espaciales cuyo SRID sea el que desea especificar.

GSE3004E No hay suficiente memoria.

Explicación: No había suficiente memoria disponible. DB2 Spatial Extender necesita hasta un máximo de 1 megabyte de memoria.

Respuesta del usuario: Libere memoria para DB2 Spatial Extender. Si no puede liberar memoria, añada más memoria física.

GSE3005E Los SRID de las geometrías difieren.

Explicación: Las geometrías que se pasan a una función de DB2 Spatial Extender deben tener el mismo identificador de sistema de referencias espaciales (SRID).

Respuesta del usuario: Vuelva a crear una de las geometrías para que su sistema de referencias espaciales coincida con el de la otra geometría.

GSE3006E Cadena binaria no válida.

Explicación: Se ha proporcionado a la función invocada una cadena binaria WKB o ESRI de formato incorrecto.

Respuesta del usuario: Reconstruya la cadena utilizando el formato correcto. Para conocer el formato correcto, consulte el manual ″DB2 Spatial Extender, Guía y consulta del usuario″.

GSE3007E No se ha especificado una geometría válida.

Explicación: No se ha proporcionado un tipo válido de geometría a la función invocada. Los tipos válidos de geometría son: punto, línea, polígono, multipunto, multilínea y multipolígono.

Respuesta del usuario: Vuelva a someter la sentencia de SQL utilizando un tipo válido de geometría.

GSE3008E Paréntesis no emparejados.

Explicación: El número de paréntesis de apertura de la cadena de texto WKT no es igual al número de paréntesis de cierre.

Respuesta del usuario: Vuelva a escribir la cadena, con igual número de paréntesis de apertura y de cierre.

GSE3009E Demasiadas partes especificadas.

Explicación: El número de partes indicadas en la cadena binaria o de texto es mayor que el número real de partes proporcionadas.

Respuesta del usuario: Vuelva a escribir la cadena utilizando el número correcto de partes.

GSE3010E Tipo de geometría no válido.

Explicación: Se ha proporcionado un tipo no válido de geometría a la función invocada. Por ejemplo, se puede haber proporcionado una línea a una función que utiliza polígonos como datos de entrada.

Respuesta del usuario: Proporcione a la función un tipo de geometría que la función pueda procesar, o utilice una función que acepte el tipo de geometría que desea proporcionar.

GSE3011E Cadena de texto demasiado larga.

Explicación: La cadena de texto de la geometría excede la longitud máxima de 4000 caracteres.

Respuesta del usuario: La geometría contiene demasiado detalle para convertirla a texto. Si embargo, como alternativa puede convertirla al formato WKB o al formato binario de forma ESRI.

GSE3012E Valor de parámetro no válido.

Explicación: Se ha proporcionado un parámetro no válido a la función.

Respuesta del usuario: Compare la sintaxis de la función con la sintaxis descrita en el manual ″DB2 Spatial Extender, Guía y consulta del usuario″. Corrija el parámetro no válido y especifique de nuevo la función.

GSE3013E Tamaño de cuadrícula no válido.

Explicación: Se ha efectuado una de las siguientes especificaciones no válidas:

- Ha especificado un número negativo como tamaño de cuadrícula para primero, segundo o tercer nivel de cuadrícula.
- Ha especificado 0 como tamaño de cuadrícula para el primer nivel de cuadrícula.
- El tamaño de cuadrícula especificado para el segundo nivel de cuadrícula es menor que el tamaño de cuadrícula del primer nivel.
- El tamaño de cuadrícula especificado para el tercer nivel de cuadrícula es menor que el tamaño de cuadrícula del segundo nivel.

Respuesta del usuario: Utilice la ventana Crear Índice o el procedimiento almacenado db2gse.gse_enable_idx para especificar un tamaño válido de cuadrícula. Para conocer los tamaños correctos de la cuadrícula, consulte el manual ″DB2 Spatial Extender, Guía y consulta del usuario″.

GSE3014E Tamaño de cuadrícula demasiado pequeño.

Explicación: El tamaño de cuadrícula especificado da lugar a más de 1000 celdas de cuadrícula para cada geometría.

Respuesta del usuario: Utilice la ventana Crear Índice o el procedimiento almacenado db2gse.gse_enable_idx para aumentar el tamaño de cuadrícula o añadir otro nivel de cuadrícula.

GSE3015E Se ha creado una geometría no válida.

Explicación: Los parámetros entrados han producido una geometría no válida. Por ejemplo, los parámetros entrados con la función LineFromShape producen una geometría no válida. Una geometría no válida es la que vulnera una propiedad de geometría.

Respuesta del usuario: Corrija el parámetro y especifique de nuevo la geometría.

GSE3016E Se han proporcionado geometrías incorrectas.

Explicación: La función esperaba dos geometrías de un tipo determinado y no las ha recibido. Por ejemplo, la función ST_Union espera dos geometrías de la misma dimensión y ha recibido un punto y una línea, que tienen dimensiones diferentes.

Respuesta del usuario: Especifique geometrías que la función acepte como entrada válida. Para conocer los tipos válidos de geometría utilizados por la función, consulte el manual ″DB2 Spatial Extender, Guía y consulta del usuario″.

GSE3017E Error de integridad de geometría.

Explicación: La función no puede procesar la geometría proporcionada, pues una o más propiedades de la geometría vulneran una restricción de integridad.

Respuesta del usuario: Vuelva a especificar la geometría, con sus propiedades definidas correctamente. Para obtener información sobre las propiedades de las geometrías, consulte el manual ″DB2 Spatial Extender, Guía y consulta del usuario″.

GSE3018E Demasiados puntos.

Explicación: La creación de una geometría ha excedido el límite de almacenamiento de 1 MB; la geometría tiene demasiados puntos.

Respuesta del usuario: Elimine puntos innecesarios. Por razones de rendimiento y espacio de almacenamiento, incluya sólo los puntos necesarios para representar una geometría. Todos los puntos no esenciales se deben excluir.

GSE3019E Geometría demasiado pequeña.

Explicación: La geometría devuelta por la función ST_Difference, ST_Intersection, ST_SymmetricDiff o ST_Union es demasiado pequeña para poder ser representada mediante valores del sistema actual de coordenadas.

Respuesta del usuario: Si es necesario disponer de un resultado, utilice el procedimiento almacenado db2gse.gse_enable_sref para aumentar el parámetro xyunits del sistema de referencias espaciales de la geometría de origen. Luego, reconstruya la tabla donde se guarda la geometría de origen.

GSE3020E La distancia de rodeo está fuera de los límites.

Explicación: La función BUFFER ha creado una distancia de rodeo que queda fuera del sistema de coordenadas.

Respuesta del usuario: Disminuya la distancia de rodeo o modifique el sistema de coordenadas de la geometría de origen. En la mayoría de los casos, para modificar el sistema de coordenadas es necesario volver a cargar el sistema de referencias espaciales.

GSE3021E Factor de escala no válido.

Explicación: El factor de escala (una unidad XY, una unidad Z o una unidad M) no puede ser menor que 1.

Respuesta del usuario: Utilice el procedimiento almacenado db2gse.gse_enable_sref para corregir los factores de escala contenidos en la vista de catálogo DB2GSE.SPATIAL_REF_SYS que sean menores que 1.

GSE3022E Coordenada fuera de los límites.

Explicación: Una coordenada es demasiado grande o demasiado pequeña para quedar dentro de los límites del sistema de coordenadas.

Respuesta del usuario: Determine si la coordenada es correcta. Si es correcta, determine si queda dentro de los límites del sistema de coordenadas que está utilizando. Para obtener información sobre este sistema de coordenadas, consulte la vista de catálogo DB2GSE.COORD_REF_SYS.

GSE3023E ID de sistema de coordenadas no válido.

Explicación: Spatial Extender no puede validar el identificador de sistema de coordenadas especificado.

Respuesta del usuario: Determine si el identificador aparece listado en la vista de catálogo DB2GSE.COORD_REF_SYS. Si no está listado, verifique que sea correcto y solicite al administrador de la base de datos que lo registre en el catálogo del sistema de Spatial Extender.

GSE3024E Texto de comentario no válido.

Explicación: El texto de comentario que sirve para definir el sistema de coordenadas especificado no se puede convertir en una proyección válida.

Respuesta del usuario: Busque el texto de comentario correspondiente a este sistema de coordenadas en la vista de catálogo DB2GSE.COORD_REF_SYS. Determine si el texto define el sistema correctamente. El manual "DB2 Spatial Extender, Guía y consulta del usuario" contiene información útil en su capítulo sobre sistemas de coordenadas.

GSE3025E Error de proyección.

Explicación: Se ha producido un error al intentar proyectar una geometría.

Respuesta del usuario: Compruebe que la geometría está dentro del dominio permitido de la proyección.

GSE3026E Los anillos de un polígono se solapan.

Explicación: Los anillos de un polígono no se pueden solapar, pero pueden formar una intersección en un punto de tangencia.

Respuesta del usuario: Corrija las coordenadas del polígono y especifíquelo otra vez.

GSE3027E Demasiados pocos puntos.

Explicación: Una línea consta como mínimo de dos puntos, y un polígono debe tener como mínimo cuatro puntos.

Respuesta del usuario: Vuelva a especificar la geometría utilizando el número correcto de puntos.

GSE3028E El polígono no está cerrado.

Explicación: Las coordenadas de los puntos inicial y final del polígono no son iguales.

Respuesta del usuario: Edite la lista de coordenadas del polígono para que los puntos inicial y final coincidan, luego especifique de nuevo el polígono.

GSE3029E El anillo exterior no es válido.

Explicación: El anillo exterior no encierra al anillo interior. El anillo interior está completamente fuera del anillo exterior, sin existir ningún solapamiento.

Respuesta del usuario: Compruebe que las coordenadas del anillo interior quedan completamente dentro del anillo exterior. Si el anillo interior en realidad representa el anillo exterior de otro polígono, especifique la geometría como multipolígono.

GSE3030E El polígono carece de superficie.

Explicación: Una geometría es un polígono sólo si sus coordenadas abarcan dos dimensiones en el espacio.

Respuesta del usuario: Edite las coordenadas del polígono para que delimiten una superficie y especifique de nuevo el polígono. Como alternativa, especifique una línea, si es apropiado.

GSE3031E El polígono contiene un bucle.

Explicación: Sólo el punto inicial y final de un polígono pueden ser iguales. Todas las demás coordenadas de un anillo de polígono deben ser diferentes y en su conjunto delimitar una superficie.

Respuesta del usuario: Busque pares de coordenadas que tengan los mismos valores X e Y. Edite estos puntos para que el polígono encierre una sola superficie; luego, especifique de nuevo el polígono.

GSE3032E Los anillos exteriores se solapan.

Explicación: Los anillos exteriores de un multipolígono pueden formar una intersección en un punto de tangencia, pero no se pueden solapar.

Respuesta del usuario: Edite las coordenadas de los anillos exteriores para que no se solapen; luego, especifique de nuevo el multipolígono.

GSE3033E El polígono forma una intersección consigo mismo.

Explicación: El anillo de un polígono no puede formar una intersección consigo mismo.

Respuesta del usuario: Edite las coordenadas del anillo que forma una intersección consigo mismo; luego, especifique de nuevo el polígono.

GSE3034E Número no válido de medidas.

Explicación: El parámetro *número de medidas* de la cadena binaria especifica un número de medidas que es diferente del número de medidas proporcionadas con la cadena.

Respuesta del usuario: Edite el parámetro *número de medidas* para que se corresponda con el número de medidas proporcionadas en la cadena binaria.

GSE3035E Número no válido de partes.

Explicación: El parámetro *número de partes* de la cadena binaria especifica un número de partes que es diferente del número de partes proporcionadas con la cadena.

Respuesta del usuario: Edite el parámetro *número de partes* para que se corresponda con el número de partes proporcionadas en la cadena binaria.

GSE3036E Desplazamiento de parte no válido.

Explicación: El parámetro *desplazamiento de parte* de la cadena binaria especifica un desplazamiento de parte que es diferente del desplazamiento de parte proporcionado dentro de la cadena.

Respuesta del usuario: Edite el parámetro *desplazamiento de parte* para que se corresponda con el desplazamiento de parte proporcionado en la cadena binaria.

GSE3037E Error de proyección.

Explicación: Se ha encontrado una geometría no permitida; su separador de partes no es válido.

Respuesta del usuario: Consulte al representante de servicio de IBM.

GSE3038E BLOB demasiado pequeño.

Explicación: El número de bytes del gran objeto binario (BLOB) especificado es menor que el número de bytes del BLOB proporcionado.

Respuesta del usuario: Haga que la longitud del BLOB sea igual al número de bytes del BLOB; luego, especifique de nuevo la función.

GSE3039E Tipo de entidad no válido.

Explicación: Se ha encontrado una geometría no permitida; su tipo de entidad correspondiente no es válido.

Respuesta del usuario: Consulte al representante de servicio de IBM.

GSE3040E Orden de bytes no válido.

Explicación: El orden de bytes debe ser 0 ó 1.

Respuesta del usuario: Edite el orden de bytes para que sea 0 (″little endian″) o 1 (″big endian″).

GSE3041E Parte no válida.

Explicación: Un parámetro de función ha indexado una parte que no existe. Por ejemplo, se produciría este error si se pasa un 3 a la función ST_GeometryN para que devuelva el tercer punto de un multipunto, y éste sólo contiene dos puntos.

Respuesta del usuario: Edite el parámetro y especifique de nuevo la función.

GSE3042E Geometría vacía.

Explicación: Se ha pasado una geometría vacía a la función ST_AsBinary, pero una geometría vacía no es un dato de entrada válido para la función.

Respuesta del usuario: Edite la sentencia de SQL especificada para que sólo se proporcione una geometría no vacía a la función ST_AsBinary. Por ejemplo, puede utilizar una cláusula WHERE para descartar geometrías vacías mediante la función ST_IsEmpty.

Cuando una función espacial devuelve un mensaje, DB2 visualiza la forma abreviada del mensaje y el correspondiente valor SQLSTATE dentro del mensaje SQL0443N. He aquí un ejemplo:

```
DB21034E  El mandato se ha procesado como sentencia de SQL porque no
era un mandato válido del Procesador de Línea de Mandatos. Durante el
proceso de SQL, el mandato devolvió:
SQL0443N  La rutina "DB2GSE.ST_POINTFROMTEX" (nombre específico
"SQL000503150228187") ha devuelto un SQLSTATE de error con el texto de
diagnóstico "SRID no válido".  SQLSTATE=38601
```

Para determinar el SQLCODE asociado al SQLSTATE devuelto en el mensaje SQL0443N, consulte la Tabla 34. Para ver texto completo correspondiente al SQLCODE, consulte el presente capítulo o emita el mandato siguiente:

```
DB2 ? [SQLCODE]
```

Tabla 34. Valores SQLSTATE y valores SQLCODE de los mensajes devueltos por las funciones espaciales

Si el valor SQLSTATE es este:	. . . el valor SQLCODE es este:
38600	GSE3002E
38601	GSE3003E
38602	GSE3004E
38603	GSE3005E
38604	GSE3006E
38605	GSE3007E
38606	GSE3008E

Tabla 34. Valores SQLSTATE y valores SQLCODE de los mensajes devueltos por las funciones espaciales (continuación)

Si el valor SQLSTATE es este:	. . . el valor SQLCODE es este:
38607	GSE3009E
38608	GSE3010E
38609	GSE3011E
38610	GSE3012E
38612	GSE3013E
38613	GSE3014E
38800	GSE3015E
38801	GSE3016E
38802	GSE3017E
38803	GSE3018E
38804	GSE3019E
38805	GSE3020E
38806	GSE3021E
38807	GSE3022E
38808	GSE3023E
38809	GSE3024E
38810	GSE3025E
38811	GSE3026E
38812	GSE3027E
38813	GSE3028E
38814	GSE3029E
38815	GSE3030E
38816	GSE3031E
38817	GSE3032E
38818	GSE3033E
38819	GSE3034E
38820	GSE3035E
38821	GSE3036E
38822	GSE3037E
38823	GSE3038E
38824	GSE3039E

Tabla 34. Valores SQLSTATE y valores SQLCODE de los mensajes devueltos por las funciones espaciales (continuación)

Si el valor SQLSTATE es este:	**. . . el valor SQLCODE es este:**
38825	GSE3040E
38826	GSE3041E
38827	GSE3042E
38999	GSE3043E

Capítulo 11. Vistas de catálogo

Las vistas de catálogo de Spatial Extender contienen metadatos sobre:

- Sistemas de coordenadas que puede utilizar. Para obtener información como textos de anotación e identificadores de estos sistemas, consulte la sección "DB2GSE.COORD_REF_SYS".
- Columnas espaciales que se han registrado como capas. Para obtener información como los nombres, los tipos de datos y los sistemas de referencias espaciales asociados a estas columnas, consulte la sección "DB2GSE.GEOMETRY_COLUMNS" en la página 146.
- Geocodificadores que puede utilizar. Para obtener información como los identificadores de estos geocodificadores y las regiones que contienen las ubicaciones que procesan los geocodificadores, consulte la sección "DB2GSE.SPATIAL_GEOCODER" en la página 147.
- Sistemas de referencias espaciales que puede utilizar. Para obtener información como sus identificadores y descripciones, consulte la sección "DB2GSE.SPATIAL_REF_SYS" en la página 147.

DB2GSE.COORD_REF_SYS

Cuando habilita una base de datos para operaciones espaciales, Spatial Extender registra los sistemas de coordenadas que puede utilizar en una tabla de catálogo. Las columnas seleccionadas de esta tabla forman la vista de catálogo DB2GSE.COORD_REF_SYS, que se describe en la Tabla 35.

Tabla 35. Columnas de la vista de catálogo DB2GSE.COORD_REF_SYS

Nombre	Tipo de datos	¿Posibilidad de nulos?	Contenido
CSID	INTEGER	No	Identificador numérico exclusivo para este sistema de coordenadas.
CS_NAME	VARCHAR(64)	No	Nombre de este sistema de coordenadas.
AUTH_NAME	VARCHAR(256)	Sí	Sistema de coordenadas con el que cumple la organización que ha compilado este sistema de coordenadas; por ejemplo, European Petroleum Survey Group (EPSG).
AUTH_SRID	INTEGER	Sí	Un identificador numérico asignado a este sistema de coordenadas por la organización especificada en la columna AUTH_NAME.
DESC	VARCHAR(256)	Sí	Descripción de este sistema de coordenadas.

Tabla 35. Columnas de la vista de catálogo DB2GSE.COORD_REF_SYS (continuación)

Nombre	Tipo de datos	¿Posibilidad de nulos?	Contenido
SRTEXT	VARCHAR(2048)	No	Texto de anotación para este sistema de coordenadas.

DB2GSE.GEOMETRY_COLUMNS

Cuando crea una capa, Spatial Extender la registra grabando su identificador e información relacionada con la misma en la tabla de catálogo. Las columnas seleccionadas de esta tabla forman la vista de catálogo DB2GSE.GEOMETRY_COLUMNS, que se describe en la Tabla 36.

Tabla 36. Columnas de la vista de catálogo DB2GSE.GEOMETRY_COLUMNS

Nombre	Tipo de datos	¿Posibilidad de nulos?	Contenido
LAYER_CATALOG	VARCHAR(30)	Sí	NULL. El concepto LAYER_CATALOG no existe en Spatial Extender.
LAYER_SCHEMA	VARCHAR(30)	No	Esquema de la tabla o vista que contiene la columna que se ha registrado como esta capa.
LAYER_TABLE	VARCHAR(128)	No	Nombre de la tabla o vista que contiene la columna que se ha registrado como esta capa.
LAYER_COLUMN	VARCHAR(128)	No	Nombre de la columna que se ha registrado como esta capa.
GEOMETRY_TYPE	INTEGER	No	Tipo de datos de la columna que se ha registrado como esta capa.
SRID	INTEGER	No	Identificador del sistema de referencias espaciales utilizado para los valores de la columna que se ha registrado como esta capa.
STORAGE_TYPE	INTEGER	Sí	Información sobre cómo DB2 almacena los valores de la columna que se ha registrado como esta capa. Por ejemplo, los datos de STORAGE_TYPE pueden indicar que los valores se almacenan como objetos grandes (LOB).

DB2GSE.SPATIAL_GEOCODER

Los geocodificadores disponibles se registran en una tabla de catálogo. Las columnas seleccionadas de esta tabla forman la vista de catálogo DB2GSE.SPATIAL_GEOCODER, que se describe en la Tabla 37.

Tabla 37. Columnas de la vista de catálogo DB2GSE.SPATIAL_GEOCODER

Nombre	Tipo de datos	¿Posibilidad de nulos?	Contenido
GCID	INTEGER	No	Identificador numérico del geocodificador.
GC_NAME	VARCHAR(64)	No	Identificador alfabético del geocodificador.
VENDOR_NAME	VARCHAR(128)	No	Nombre del proveedor que suministró el geocodificador.
PRIMARY_UDF	VARCHAR(256)	No	Nombre completo del geocodificador.
PRECISION_LEVEL	INTEGER	No	El grado en que los datos fuente deben coincidir con los datos de referencia correspondientes para que el geocodificador los procese satisfactoriamente.
VENDOR_SPECIFIC	VARCHAR(256)	Sí	Nombre y vía de acceso de un archivo que puede utilizar un proveedor para definir parámetros especiales a los que da soporte el geocodificador.
GEO_AREA	VARCHAR(256)	Sí	Zona geográfica que contiene los lugares que se deben geocodificar.
DESCRIPTION	VARCHAR(256)	Sí	Descripción del geocodificador.

DB2GSE.SPATIAL_REF_SYS

Cuando crea un sistema de referencias espaciales, Spatial Extender lo registra grabando su identificador e información relacionada con el mismo en una tabla de catálogo. Las columnas seleccionadas de esta tabla forman la vista de catálogo DB2GSE.SPATIAL_REF_SYS, que se describe en la Tabla 38.

Tabla 38. Columnas de la vista de catálogo DB2GSE.SPATIAL_REF_SYS

Nombre	Tipo de datos	¿Posibilidad de nulos?	Contenido
SRID	INTEGER	No	Identificador definido por el usuario de este sistema de referencias espaciales.
SR_NAME	VARCHAR(64)	No	Nombre de este sistema de referencias espaciales.
CSID	INTEGER	No	Identificador numérico del sistema de coordenadas asociado al sistema de referencias espaciales.

Tabla 38. Columnas de la vista de catálogo DB2GSE.SPATIAL_REF_SYS (continuación)

Nombre	Tipo de datos	¿Posibilidad de nulos?	Contenido
CS_NAME	VARCHAR(64)	No	Nombre del sistema de coordenadas asociado al sistema de referencias espaciales.
AUTH_NAME	VARCHAR(256)	Sí	Nombre de la organización que define los estándares correspondientes a este sistema de referencias espaciales.
AUTH_SRID	INTEGER	Sí	El identificador que la organización especificada en la columna AUTH_NAME asigna a este sistema de referencias espaciales.
SRTEXT	VARCHAR(2048)	No	Texto de anotación correspondiente a este sistema de referencias espaciales.
FALSEX	FLOAT	No	Un número que, al restarse de un valor negativo de coordenada X, da como resultado un número no negativo (es decir, un número positivo o un cero).
FALSEY	FLOAT	No	Un número que, al restarse de un valor negativo de coordenada Y, da como resultado un número no negativo (es decir, un número positivo o un cero).
XYUNITS	FLOAT	No	Un número que, cuando se multiplica por una coordenada X decimal o por una coordenada Y decimal, da como resultado un entero que se puede almacenar como dato de 32 bits.
FALSEZ	FLOAT	No	Un número que, al restarse de un valor negativo de coordenada Z, da como resultado un número no negativo (es decir, un número positivo o un cero).
ZUNITS	FLOAT	No	Un número que, cuando se multiplica por una coordenada Z decimal, da como resultado un entero que se puede almacenar como dato de 32 bits.
FALSEM	FLOAT	No	Un número que, al restarse de una medida negativa, da como resultado un número no negativo (es decir, un número positivo o un cero).
MUNITS	FLOAT	No	Un número que, cuando se multiplica por una medida decimal, da como resultado un entero que se puede almacenar como dato de 32 bits.

Capítulo 12. Índices espaciales

Puesto que las columnas espaciales contienen datos geográficos de dos dimensiones, las aplicaciones que efectúan consultas en dichas columnas necesitan una estrategia de índice que identifique rápidamente todas las geometrías contenidas en una determinada extensión. Por este motivo, Spatial Extender ofrece el índice espacial de tres niveles basado en una cuadrícula.

Este capítulo describe este tipo de índice y contiene directrices para utilizarlo. Los temas de esta sección son:

Un fragmento de programa de ejemplo

Utilizaremos el siguiente ejemplo para ver cómo se crea un índice y se utiliza en SQL. Consulte el manual *SQL Reference* para obtener más información sobre los mandatos CREATE INDEX y CREATE INDEX EXTENSION. Tenga en cuenta que, una vez creado el índice, puede emitir sentencias DDL y DML estándares que utilicen predicados y funciones espaciales.

```
create table customers (cid int, addr varchar(40), ..., loc db2gse.ST_Point)
create table stores (sid int, addr varchar(40), ..., loc db2gse.ST_Point,
 zone db2gse.ST_Polygon)

create index customersx1 on customers(loc) extend using db2gse.spatial_index(10e0,
 100e0, 1000e0)
create index storesx1 on stores(loc) extend using db2gse.spatial_index(10e0, 100e0,
 1000e0)
create index storesx2 on stores(zone) extend using db2gse.spatial_index(10e0, 100e0,
 1000e0)

insert into customers (cid, addr, loc) values
(:cid, :addr, sdeFromBinary(:loc))
insert into customers (cid, addr, loc) values
(:cid, :addr, geocode(:addr))
insert into stores (sid, addr, loc) values
(:sid, :addr, sdeFromBinary(:loc))

update stores set zone = db2gse.ST_Buffer (loc, 2)

select cid, loc from customers
 where db2gse.ST_Within(loc, :polygon) = 1

select cid, loc from customers
 where db2gse.ST_Within(loc, :circle1) = 1 OR
       db2gse.ST_Within(loc, :circle2) = 1
```

```
select c.cid, loc from customers c, stores s
 where db2gse.ST_Contains(s.zone, c.loc) = 1 selectivity 0.01

select avg(c.income) from customers c
 where not exist (select * from stores s
                  where db2gse.ST_Distance(c.loc, s.loc) < 10)
```

Índices de árbol B

La tecnología de indexación espacial se basa en el índice de árbol B jerárquico tradicional, pero es significativamente diferente. El índice espacial utiliza *indexación de cuadrícula* que está diseñada para indexar columnas espaciales de dos dimensiones. El índice de árbol B sólo puede manejar datos de una dimensión y no se puede utilizar con información GIS. Esta sección describe cómo se estructura y se utiliza un índice de árbol B.

El nivel superior de un índice de árbol B, denominado nodo raíz, contiene una clave para cada nodo del siguiente nivel. El valor de cada clave es el valor de clave mayor existente para el nodo correspondiente del siguiente nivel. En función del número de valores de la tabla base, es posible que se necesiten varios nodos intermedios. Estos nodos forman un puente entre el nodo raíz y los nodos hoja que alojan los ID reales de filas de la tabla base.

El gestor de la base de datos busca en un índice de árbol B empezando por el nodo raíz. Luego continúa por los nodos intermedios hasta alcanzar el nodo hoja con el ID de fila de la tabla base.

El índice de árbol B no se puede aplicar a una columna espacial porque la característica de dos dimensiones de la columna espacial necesita la estructura de un índice espacial. Por el mismo motivo, no puede aplicar un índice espacial a una columna que no sea espacial. Además, un índice espacial no se puede crear a partir de varias columnas.

Formas de crear un índice espacial

Hay varias formas de crear un índice espacial:

- Definiendo uno en la ventana Crear índice espacial. Para obtener instrucciones, consulte el “Capítulo 6. Creación de índices espaciales” en la página 67.
- Invocando el procedimiento almacenado db2gse.gse_enable_idx en un programa de aplicación. Para obtener información sobre este procedimiento almacenado, consulte el “Capítulo 9. Procedimientos almacenados” en la página 83.
- Emitiendo el mandato **db2 create index** con la función **spatial_index** en la cláusula USING, Por ejemplo:

```
create index storesx1 on customers (loc) using db2gse.spatial_index(10e0,
100e0, 1000e0)
```

La naturaleza de los datos espaciales requiere que el diseñador de la base de datos comprenda su distribución relativa de tamaños. El diseñador debe determinar el tamaño óptimo y el número de niveles de cuadrícula con los que crear el índice espacial.

Los niveles de cuadrícula, <nivel de cuadrícula 1>, <nivel de cuadrícula 2> y <nivel de cuadrícula 3>, se especifican aumentando el tamaño de celda. Por lo tanto, el segundo nivel debe tener un tamaño de celda superior al del primero, y el tercero uno superior al del segundo. El primer nivel de cuadrícula es obligatorio, pero puede inhabilitar el segundo y el tercer nivel con un valor cero de doble precisión (0.0e0).

Cómo se genera un índice espacial

Un índice espacial se genera utilizando *envolventes*. La envolvente es una geometría y representa la extensión X e Y mínima y máxima de una geometría. Para la mayoría de las geometrías, la envolvente es un recuadro, pero para líneas horizontales y verticales, la envolvente es una línea de dos puntos. Para puntos, la envolvente es un punto. Para obtener más información sobre envolventes, consulte la sección “Envolvente” en la página 165.

El índice espacial se construye en una columna espacial creando una o más entradas correspondientes a las intersecciones de cada envolvente de geometría con la cuadrícula. Una intersección se registra como el ID interno de la geometría y las coordenadas X e Y mínimas de la celda de la cuadrícula que forma la intersección. Por ejemplo, el polígono de la Figura 7 en la página 152 forma intersección con la cuadrícula en las coordenadas (20,30), (30,30), (40,30), (20,40), (30,40), (40,40), (20,50), (30,50) y (40,50). Consulte la Tabla 39 en la página 152 para ver las coordenadas X e Y mínimas para todas las geometrías de la Figura 7 en la página 152.

Si hay varios niveles de cuadrícula, Spatial Extender intenta utilizar el nivel de cuadrícula más bajo posible. Cuando una geometría forma intersección con cuatro o más celdas de cuadrícula a un determinado nivel, se promociona al nivel superior siguiente. Por lo tanto, para un índice espacial que tiene los tres niveles de cuadrícula de 10.0e0, 100.0e0 y 1000.0e0, Spatial Extender formará primero intersección entre cada geometría con la cuadrícula de nivel 10.0e0. Si una geometría forma intersección con cuatro o más celdas de cuadrícula 10.0e0, se promociona y formará intersección con la cuadrícula de nivel 100.0e0. Si hay cuatro o más intersecciones al nivel 100.0e0, la geometría se promociona al nivel 1000.0e0. En el nivel 1000.0e0, las intersecciones se deben entrar en el índice espacial porque es el nivel más alto posible.

La Figura 7 ilustra cómo cuatro tipos diferentes de geometrías forman intersección con una cuadrícula 10.0e. Las 23 intersecciones correspondientes a las cuatro geometrías se registran en el índice espacial.

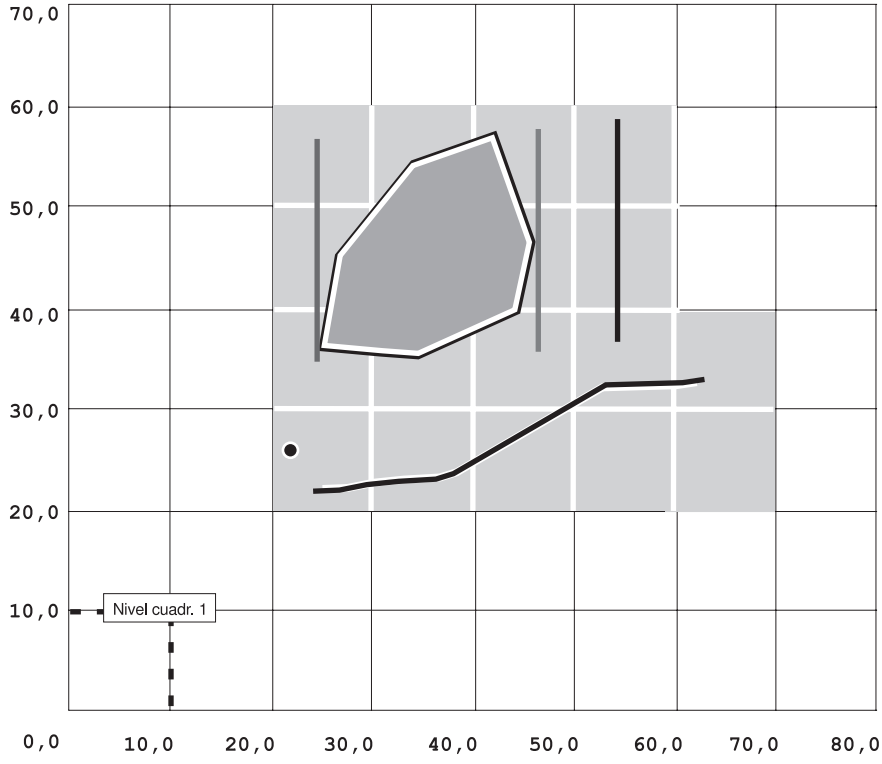


Figura 7. Aplicación a un nivel de cuadrícula 10.0e0

La Tabla 39 muestra las geometrías y sus correspondientes intersecciones de cuadrícula. Las envolventes de cuatro tipos de geometrías diferentes forman intersección con la cuadrícula 10.0e. La coordenada X e Y mínima de cada celda de cuadrícula que forma intersección se entra en el índice espacial.

Tabla 39. Entradas de celda de cuadrícula 10.0e0 correspondientes a las geometrías de ejemplo

Geometría	Cuadrícula X	Cuadrícula Y
Polígono	20,0	30,0
Polígono	30,0	30,0
Polígono	40,0	30,0
Polígono	20,0	40,0

Tabla 39. Entradas de celda de cuadrícula 10.0e0 correspondientes a las geometrías de ejemplo (continuación)

Geometría	Cuadrícula X	Cuadrícula Y
Polígono	30,0	40,0
Polígono	40,0	40,0
Polígono	20,0	50,0
Polígono	30,0	50,0
Polígono	40,0	50,0
Línea vertical	50,0	30,0
Línea vertical	50,0	40,0
Línea vertical	50,0	50,0
Punto	20,0	20,0
Línea horizontal	20,0	20,0
Línea horizontal	30,0	20,0
Línea horizontal	40,0	20,0
Línea horizontal	50,0	20,0
Línea horizontal	60,0	20,0
Línea horizontal	20,0	30,0
Línea horizontal	30,0	30,0
Línea horizontal	40,0	30,0
Línea horizontal	50,0	30,0
Línea horizontal	60,0	30,0

La Figura 8 en la página 154 muestra cómo el número de intersecciones se reduce significativamente a ocho al añadir los niveles de cuadrícula 30.0e0 y 60.0e0. En este caso, el polígono identificado como geometría 1 se promociona al nivel de cuadrícula 30.0e0 y la línea identificada como geometría 4 se promociona al nivel de cuadrícula 60.0e0. En lugar de las nueve y diez intersecciones que tenían las geometrías en el nivel 10.0e0, sólo tienen dos tras la promoción.

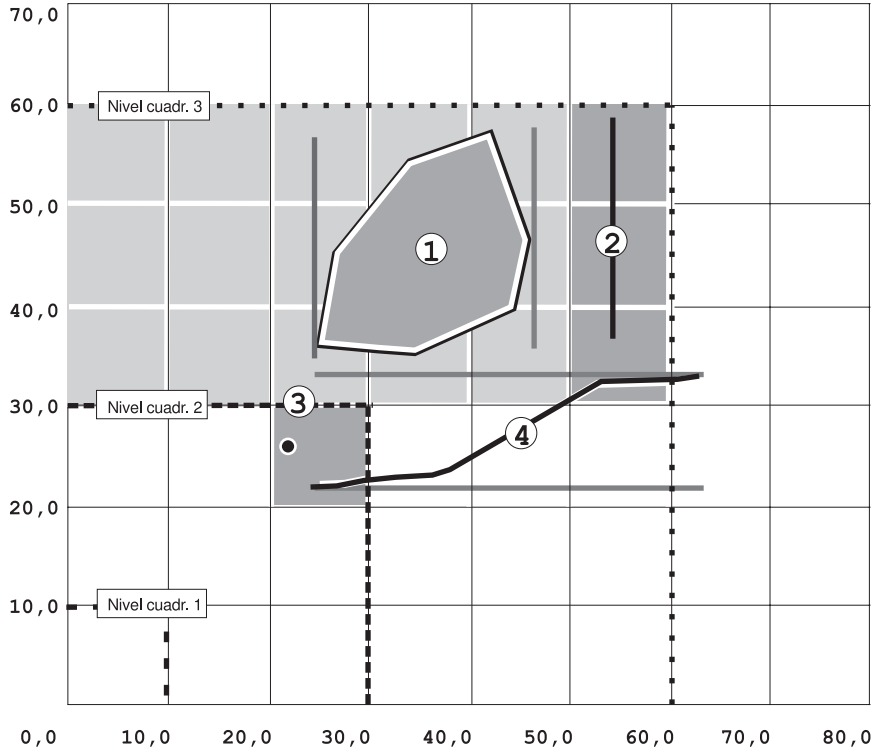


Figura 8. Efecto de añadir los niveles de cuadrícula 30.0e0 y 60.0e0. La envolvente del polígono identificado como geometría 1 forma intersección con nueve celdas de cuadrícula. La envolvente de la línea vertical identificada como geometría 2 forma intersección con tres celdas de cuadrícula. La envolvente del punto identificado como geometría 3 forma intersección con una sola celda de cuadrícula. La envolvente de la línea identificada como geometría 4 forma intersección con diez celdas de cuadrícula.

Spatial Extender toma los parámetros de nivel de cuadrícula especificados en la sentencia CREATE INDEX y comprueba cada objeto espacial para determinar las coordenadas y el número de bloques de cuadrícula en los que se encuentra cada objeto. En la Figura 8, los niveles de cuadrícula 10.0e0, 30.0e0 y 60.0e0 se muestran con pesos lineales crecientes y diferentes tonos de gris. Las intersecciones de las celdas de las envolventes de la línea vertical y del punto se entran en el índice al nivel de cuadrícula 10.0e0, porque ambas generan menos de cuatro intersecciones. El polígono forma intersección con nueve celdas de la cuadrícula 10.0e0, por lo que se promociona al nivel de cuadrícula 30.0e0. A este nivel, el polígono forma intersección con dos celdas de cuadrícula, que se entran en el índice. La línea identificada como geometría 4 forma intersección con diez celdas de la cuadrícula 10.0e0, por lo que se promociona al nivel de cuadrícula 30.0e0. Ya a este nivel, forma intersección con seis celdas de cuadrícula, por lo que se vuelve a promocionar al nivel de cuadrícula 60.0e0, donde genera dos intersecciones. Las intersecciones de la cuadrícula 60.0e0 de la línea se entran en el índice. Si la línea hubiera

generado cuatro o más intersecciones a este nivel, se entraría igualmente en el índice porque es el nivel superior al que se puede promocionar una geometría.

Tabla 40. Las intersecciones de las geometrías en el índice de tres niveles

Geometría	Cuadrícula X	Cuadrícula Y
Las intersecciones entre la línea vertical y el punto en el nivel 1 (tamaño de cuadrícula 10.0e0)		
2	50,0	30,0
2	50,0	40,0
2	50,0	50,0
3	20,0	20,0
Las intersecciones del polígono en el nivel 2 (tamaño de cuadrícula 30.0e0)		
1	0,0	30,0
1	30,0	30,0
Las intersecciones de la línea en el nivel 3 (tamaño de cuadrícula 60.0e0)		
4	0,0	0,0
4	60,0	0,0

Spatial Extender realmente no crea ninguna estructura de cuadrículas de polígono de ningún tipo. Spatial Extender muestra cada nivel de cuadrícula de forma paramétrica definiendo el origen en el desplazamiento X,Y del sistema de referencias espaciales de las columnas. Luego amplía la cuadrícula a un espacio de coordenadas positivas. Utilizando una cuadrícula paramétrica, Spatial Extender genera las intersecciones de forma matemática.

Directrices sobre la utilización de un índice espacial

Spatial Extender trabaja con un índice espacial para mejorar el rendimiento de una consulta espacial. Considere, por ejemplo, la consulta espacial más básica y probablemente de uso más habitual: la consulta de recuadro. Esta consulta solicita a Spatial Extender que devuelva todas las geometrías que se encuentran total o parcialmente dentro de un recuadro definido por el usuario. Si no existe un índice, Spatial Extender debe comparar todas las geometrías con el recuadro. Sin embargo, con un índice Spatial Extender puede localizar todas las entradas de índice que tienen una coordenada inferior izquierda superior o igual a la del recuadro y una coordenada superior derecha inferior o igual a la del recuadro. Puesto que el índice está ordenado por el sistema de coordenadas, Spatial Extender puede obtener rápidamente una lista de geometrías candidato. El proceso que se acaba de describir recibe el nombre de *primer pase*.

Un *segundo pase* determina si la envolvente de cada candidato forma intersección con el recuadro. Una geometría que es apropiada para el primer pase, porque la envolvente de sus celdas de cuadrícula forma intersección con el recuadro, puede tener una envolvente que no lo haga.

Un *tercer pase* compara las coordenadas reales del candidato con el recuadro para determinar si cualquier parte de la geometría se encuentra realmente dentro del recuadro. Este último y más complejo proceso de comparación opera sobre una lista de candidatos compuesta por un subconjunto de la población total, la cual disminuye significativamente tras los dos primeros pases.

Todas las consultas espaciales llevan a cabo los tres pases excepto la función EnvelopesIntersect. Esta sólo realiza los dos primeros pases. La función EnvelopesIntersect se ha diseñado para operaciones de visualización que suelen utilizar sus propias rutinas de recorte gráfico incorporadas y no necesitan la granularidad del tercer pase.

Selección del tamaño de celda de cuadrícula

La forma irregular de las envolventes de las geometrías complica la selección del tamaño de celda de la cuadrícula. Debido a esta irregularidad, algunas envolventes de geometrías forman intersección con varias cuadrículas, mientras que otras caben dentro de una sola celda de cuadrícula. A la inversa, en función de la distribución espacial de los datos, algunas celdas de cuadrícula interseccionan con muchas envolventes de geometría.

Para que un índice espacial funcione bien, resulta esencial seleccionar el número y el tamaño correcto de cuadrículas. Supongamos que tenemos una columna espacial que contiene geometrías de tamaño uniforme. En este caso, un solo nivel de cuadrícula será suficiente. Comience por un tamaño de celda de cuadrícula que abarque la envolvente media de las geometrías. Mientras prueba su aplicación, puede descubrir que al aumentar el tamaño de celda de la cuadrícula mejora el rendimiento de sus consultas. Esto se debe a que cada celda de cuadrícula contiene más geometrías y el primer pase puede eliminar con mayor rapidez las geometrías que no reúnen los requisitos. Sin embargo, observará que a medida que aumenta el tamaño de la celda, el rendimiento de la operación disminuye. Esto se debe a que, en última instancia, el segundo pase tiene que manejar un mayor número de geometrías posibles.

Selección del número de niveles

Si los objetos que desea indexar tienen aproximadamente el mismo tamaño relativo, puede utilizar un solo nivel de cuadrícula. Aunque esto es cierto, no todas las columnas contendrán geometrías del mismo tamaño relativo. Normalmente, las geometrías de columnas espaciales se pueden agrupar en varios rangos de tamaño. Por ejemplo, supongamos una red de carreteras en la que las geometrías se dividen en calles, carreteras principales y autopistas.

Todas las calles tienen aproximadamente la misma longitud y se pueden agrupar en un rango de tamaño. Esto también es aplicable a las carreteras principales y autopistas. Por lo tanto, las calles, que representan un rango de tamaño, se pueden agrupar en el primer nivel de cuadrícula, las redes de carreteras en el segundo y las autopistas en el tercer nivel de cuadrícula. Otro ejemplo incluye una columna de parcelas que contiene agrupaciones de pequeñas parcelas urbanas rodeadas de parcelas rurales de mayor tamaño. En este caso, hay dos intervalos de tamaños y dos niveles de cuadrícula, uno para las pequeñas parcelas urbanas y otro para las parcelas rurales de mayor tamaño. Estas situaciones son muy comunes y requieren el uso de una cuadrícula de varios niveles.

Para seleccionar el tamaño de celda de cada nivel de cuadrícula, seleccione tamaños de celda de cuadrícula ligeramente superiores que cada intervalo de tamaños. Compruebe el índice realizando consultas sobre la columna espacial.

Cada nivel adicional necesita otra exploración de índice. Vaya aumentando y disminuyendo ligeramente los tamaños de cuadrícula si consigue con ello una mejora apreciable en el rendimiento.

Capítulo 13. Geometrías y funciones espaciales asociadas

Este capítulo trata sobre unidades de información, denominadas *geometrías*, que constan de coordenadas y simbolizan elementos geográficos. También contiene una introducción a las funciones espaciales que toman geometrías como entrada y devuelven resultados que le ayudan a analizar elementos geográficos y a mover datos espaciales entre sistemas de información geográfica. Los temas de esta sección son:

- La naturaleza de las geometrías
- Propiedades de las geometrías: funciones que devuelven información referente a las propiedades
- Geometrías replicables; funciones que operan sobre ellas
- Funciones que:
  - Muestran relaciones y comparaciones entre elementos geográficos
  - Generan geometrías
  - Convierten valores de geometría en formatos que se pueden importar y exportar

Acerca de las geometrías

La obra ″Oxford American Dictionary″ define *geometría* como “la rama de las matemáticas que trata de las propiedades y las relaciones entre líneas, ángulos, superficies y formas tridimensionales.” El 11 de agosto de 1997, el denominado Open GIS Consortium Inc. (OGC), en su publicación *Open GIS Features for ODBC (SQL) Implementation Specification*, acuñó otra definición para el término. Se seleccionó la palabra *geometría* para denotar elementos geométricos que, durante el siglo pasado o incluso antes, utilizaron los cartógrafos para trazar el mapa del mundo. Una definición muy abstracta de este nuevo significado podría ser: “punto o conjunto de puntos que representan un elemento situado en la superficie terrestre”.

En Spatial Extender, una definición *de trabajo* de geometría sería; “modelo de un elemento geográfico”. El modelo se puede expresar mediante coordenadas del elemento y, en algunos casos, mediante un símbolo visual. El modelo contiene información; por ejemplo, las coordenadas identifican la posición del elemento con respecto a puntos de referencia fijos y el símbolo indica su forma. Además, el modelo se puede utilizar para generar información; por ejemplo, la función ST_Overlaps puede utilizar como entrada las coordenadas de dos regiones próximas y devolver información sobre si las regiones se solapan o no.

Las coordenadas de un elemento geográfico simbolizado por una geometría reciben el nombre de *propiedades* de la geometría. Las geometrías pueden diferir en sus propiedades; por ejemplo:

- Un *interior* representa el contenido del elemento geográfico simbolizado por la geometría.
- Un *exterior* representa el espacio circundante del elemento geográfico.
- Un *límite* representa la demarcación donde termina el contenido y comienza el espacio circundante.

Estas y otras propiedades se describen en el tema "Propiedades y funciones asociadas" en la página 162.

Las geometrías que se pueden utilizar en Spatial Extender forman una jerarquía, que se muestra en la Figura 9. Se pueden crear réplicas de seis miembros de la jerarquía; estas geometrías se pueden expresar como símbolos visuales, que también se muestran en la figura.

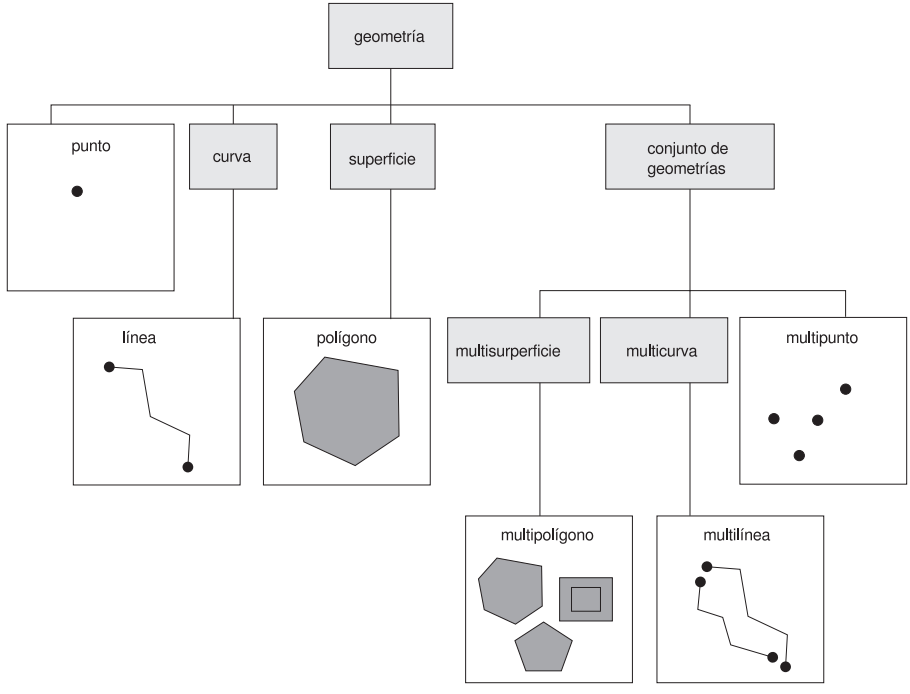


Figura 9. Jerarquía de geometrías que se pueden utilizar en Spatial Extender. Las geometrías replicables se pueden expresar como símbolos visuales. Estos símbolos se muestran debajo los nombres de esas geometrías.

Tal como indica la Figura 9 en la página 160, una superclase denominada *geometría* es la raíz de la jerarquía. Las subtipos se dividen en dos categorías: los subtipos de geometría base y los subtipos de conjunto homogéneo. Las geometrías base incluyen:

- *Puntos*, que simbolizan elementos diferenciados que se perciben como situados en el lugar geométrico donde se cruzan una línea de coordenadas este-oeste (tal como un paralelo) y una línea de coordenadas norte-sur (tal como un meridiano). Por ejemplo, supongamos que la notación en un mapa a gran escala muestra que cada ciudad del mapa se encuentra en la intersección entre un paralelo y un meridiano. En esta escala, cada ciudad se podría simbolizar mediante un punto.
- *Líneas*, que simbolizan elementos geográficos lineales (por ejemplo, calles, canales y conductos).
- *Polígonos* , que simbolizan elementos geográficos de varios lados (por ejemplo, barrios de una ciudad, bosques o hábitats de vida salvaje).

Los conjuntos homogéneos incluyen:

- *Multipuntos*, que simbolizan elementos geográficos formados por varias partes, cuyos componentes se encuentran en la intersección de una línea de coordenadas este-oeste y una línea de coordenadas norte-sur (por ejemplo, una cadena de islas cuyos miembros se encuentran situados en la intersección de un paralelo y un meridiano).
- *Multilíneas*, que simbolizan elementos geográficos formados por varios componentes o unidades lineales (por ejemplo, sistemas de ríos y sistemas de autopistas).
- *Multipolígonos*, que simbolizan elementos geográficos formados por varios componentes o unidades que tienen varios lados (por ejemplo, el grupo de granjas de una determinada región o un sistema de lagos).

Tal como indican sus nombres, los conjuntos homogéneos son agrupaciones de geometrías básicas. Además de compartir las propiedades de las geometrías básicas, los conjuntos homogéneos tienen también sus propias propiedades.

Los tipos de datos espaciales que reciben soporte de Spatial Extender son implantaciones de las geometrías que se muestran en la Figura 9 en la página 160. Para ver una descripción de estos tipos de datos, consulte la sección “Acerca de los tipos de datos espaciales” en la página 45.

Propiedades y funciones asociadas

Esta sección describe las propiedades de las geometrías y las funciones espaciales asociadas a estas propiedades. Las propiedades son:

- La clase a la que pertenece la geometría
- Coordenadas y medidas
- El interior, el límite y el exterior de una geometría
- Coordenadas Z
- Medidas
- La cualidad de ser simple o no simple
- La cualidad de estar vacía o no estar vacía
- La envolvente de una geometría
- Dimensión
- El identificador del sistema de referencias espaciales asociado a una geometría

Clase

Cada geometría pertenece a una clase de la jerarquía que se muestra en la Figura 9 en la página 160. Tal como se indica en el tema "Acerca de las geometrías" en la página 159, se pueden crear réplicas de seis subtipos de la jerarquía: puntos, líneas, polígonos, multipuntos, multilíneas y multipolígonos. No son replicables la superclase ni otros subtipos.

La función ST_GeometryType utiliza como entrada una geometría y devuelve un identificador, de tipo serie de caracteres, correspondiente al subtipo formato de una serie de caracteres. Para obtener más información, consulte la sección "ST_GeometryType" en la página 255.

La función ST_IsValid() utiliza como parámetro de entrada un valor de ST_Geometry. La función devuelve un 1 (TRUE) si la geometría es válida y un 0 (FALSE) si la geometría no es válida. Para obtener más información, consulte la sección "ST_IsValid" en la página 276.

Coordenadas y medidas

Todas las geometrías incluyen como mínimo una coordenada X y una coordenada Y. Además, una geometría puede incluir una o más coordenadas Z y medidas. Los apartados siguientes tratan de estos temas:

- Coordenadas X e Y
- Coordenadas Z y medidas
- La función The ST_CoordDim

Coordenadas X e Y

Un *valor de coordenada X* indica una ubicación en relación a un punto de referencia este u oeste. Un *valor de coordenada Y* indica una ubicación en relación a un punto de referencia norte o sur. Para obtener más información, consulte los temas "La naturaleza de los datos espaciales" en la página 6 y "Acerca de los sistemas de coordenadas y de referencias espaciales" en la página 35.

Coordenadas Z y medidas

Este apartado trata de las coordenadas Z, las medidas y la forma en que se utilizan en DB2 Spatial Extender.

Coordenadas Z

Algunas geometrías tienen una altitud o profundidad asociadas. Cada uno de los puntos que forman la geometría de un elemento geográfico puede incluir una coordenada Z adicional que representa una altitud o profundidad con respecto a la superficie terrestre.

La función AsShape convierte un valor de geometría en una representación de forma ESRI. Si el valor de geometría incluye alguna coordenada Z o medida, ésta se conserva en la representación de forma.

La función de predicado Is3d utiliza como entrada una geometría y devuelve un 1 (TRUE) si la función tiene coordenadas Z y un 0 (FALSE) si no es así. Para obtener más información, consulte la sección "Is3d" en la página 206.

Medidas

Una medida es un valor que contiene información sobre un elemento geográfico y que se almacena junto con las coordenadas que definen la ubicación del elemento. Por ejemplo, supongamos que está representando sistemas de transporte en su GIS. Si desea que su aplicación procese valores que indiquen distancias lineales o postes indicadores, puede almacenar estos valores junto con las coordenadas que definen las ubicaciones de los sistemas. Las medidas se almacenan como números de doble precisión.

Utilización de coordenadas Z y de medidas

En DB2 Spatial Extender, las coordenadas Z y las medidas se utilizan para transmitir información a las aplicaciones, no para definir ubicaciones. (En el tema "Sistemas de coordenadas, coordenadas y medidas", del Capítulo 3, ["Preparación de recursos"], encontrará ejemplos.) En consecuencia, cuando las funciones espaciales procesan puntos que incluyen coordenadas Z, medidas, o ambas cosas, en la mayoría de casos las funciones no tienen en cuenta esos valores. En cambio, otras funciones espaciales sí que operan sobre coordenadas Z y medidas:

- Puede utilizar las funciones Is3d, IsMeasured y ST_CoordDim para determinar si una geometría incluye una coordenada Z, una medida o ambas cosas.
- Puede utilizar las funciones M y Z para determinar la medida y la coordenada Z de un punto.
- El predicado IsMeasured utiliza como entrada una geometría y devuelve un 1 (TRUE) si contiene medidas y un 0 (FALSE) en caso contrario. Para obtener más información, consulte la sección "IsMeasured" en la página 207.

La función ST_CoordDim

ST_CoordDim devuelve un valor que indica qué tipo de coordenadas tiene una geometría y si la geometría contiene también alguna medida. Este valor se denomina *dimensión de coordenadas*.

ST_CoordDim puede devolver una dimensión de coordenadas igual a 2, 3 ó 4:

- Si ST_CoordDim utiliza como entrada un punto y la función devuelve un 2, significa que el punto consta de una coordenada X y una coordenada Y. Si la función utiliza como entrada un tipo de geometría distinto del punto, el valor 2 significa que cada punto de la geometría consta de una coordenada X y una coordenada Y.
- Si ST_CoordDim utiliza como entrada un punto y la función devuelve un 3, significa que el punto consta de una coordenada X, una coordenada Y, y una coordenada Z o una medida. Si la función utiliza como entrada un tipo de geometría distinto del punto, el valor 3 significa que cada punto de la geometría consta de una coordenada X, una coordenada Y, y una coordenada Z o una medida.
- Si ST_CoordDim utiliza como entrada un punto y la función devuelve un 4, significa que el punto consta de una coordenada X, una coordenada Y, una coordenada Z y una medida. Si la función utiliza como entrada un tipo de geometría distinto del punto, el valor 4 significa que cada punto de la geometría consta de una coordenada X, una coordenada Y, una coordenada Z y una medida.

Interior, límite y exterior

Todas las geometrías ocupan una posición en el espacio definida por interior, límite y exterior. El exterior de una geometría es todo el espacio que no queda ocupado por la geometría. El límite de una geometría sirve como la interfaz entre su interior y su exterior. El interior es el espacio ocupado por la geometría. Los subtipos heredan directamente las propiedades de interior y exterior, pero la propiedad de límite difiere para cada una de ellas.

La función ST_Boundary utiliza como entrada una geometría y devuelve una geometría que representa el límite de la geometría de origen. Para obtener más información, consulte la sección "ST_Boundary" en la página 229.

Simple o no simple

Algunos subtipos de geometría (líneas, multipuntos y multilíneas) son simples o no simples. Un subtipo es simple si cumple todas las reglas topológicas impuestas en el subtipo y es no simple en caso contrario. Una línea es simple si no forma intersección con su interior. Un multipunto es simple si ninguno de sus elementos ocupa el mismo espacio de coordenadas. Una multilínea es simple si ninguno de los interiores de sus elementos forma intersección con su propio interior.

La función de predicado ST_IsSimple utiliza como entrada una geometría y devuelve un 1 (TRUE) si la geometría es simple y un 0 (FALSE) en caso contrario. Para obtener más información, consulte la sección "ST_IsSimple" en la página 275.

Vacía o no vacía

Una geometría está vacía si no tiene ningún punto. La envolvente, el límite, el interior y el exterior de una geometría vacía tienen el valor NULO. Una geometría vacía es siempre simple y puede tener coordenadas Z o medidas. Las líneas o multilíneas vacías tienen una longitud 0. Los polígonos y multipolígonos vacíos tienen un superficie 0.

La función de predicado ST_IsEmpty utiliza como entrada una geometría y devuelve un 1 (TRUE) si la geometría está vacía y 0 (FALSE) si no es así. Para obtener más información, consulte la sección "ST_IsEmpty" en la página 271.

Envolvente

La envolvente de una geometría es la geometría delimitadora formada por las coordenadas mínima y máxima (X,Y). Excepto en los casos especiales descritos a continuación, la envolvente de una geometría forma un rectángulo delimitador:

- La envolvente de un punto es el propio punto, pues sus coordenadas mínima y máxima coinciden.
- La envolvente de una línea horizontal o vertical es una línea representada por el límite (los puntos finales) de la línea original.

La función ST_Envelope utiliza como entrada una geometría y devuelve una geometría delimitadora, que representa la envolvente de la geometría original. Para obtener más información, consulte la sección "ST_Envelope" en la página 249.

Dimensión

Una geometría puede tener una dimensión de 0, 1 ó 2. Las dimensiones están definidas de la manera siguiente:

0 No tiene longitud ni superficie

1 Tiene una longitud

2 Contiene una superficie

Los subtipos punto y multipunto tienen una dimensión de cero. Un punto representa un elemento geográfico con dimensión que se puede representar con una sola coordenada, mientras que el subtipo multipunto representa datos que se deben representar mediante un grupo de coordenadas dispersas.

Los subtipos línea y multilínea tienen una dimensión de uno. Estos subtipos almacenan tramos de carretera, sistemas de ríos y afluentes, y cualquier otro elemento geográfico de naturaleza lineal.

Los subtipos polígono y multipolígono tienen una dimensión de dos. Los elementos geográficos cuyo perímetro encierra un zona definible, tales como bosques, parcelas de terreno y lagos, se pueden representar mediante el tipo de datos polígono o multipolígono.

La dimensión es importante no sólo como propiedad del subtipo, sino también para determinar la relación espacial existente entre dos elementos geográficos. La dimensión del elemento o elementos geográficos resultantes determina si la operación ha sido o no satisfactoria. Spatial Extender examina la dimensión de los elementos geográficos para determinar cómo se deben comparar.

La función ST_Dimension utiliza como entrada una geometría y devuelve su dimensión como un entero. Para obtener más información, consulte la sección "ST_Dimension" en la página 243.

Identificador del sistema de referencias espaciales

El sistema de referencias espaciales identifica la transformación de coordenadas para cada geometría.

Todos los sistemas de referencias espaciales reconocidos por la base de datos son accesibles mediante la vista de catálogo DB2GSE.SPATIAL_REF_SYS. Para obtener información sobre esta vista, consulte la sección "DB2GSE.SPATIAL_REF_SYS" en la página 147.

La función ST_SRID utiliza como entrada una geometría y devuelve su identificador de referencias espaciales en forma de número entero. Para obtener más información, consulte la sección "ST_SRID" en la página 309.

La función ST_Transform asocia una geometría a un sistema de referencias espaciales distinto del que tiene asignado actualmente la geometría. Para obtener más información, consulte la sección "ST_Transform" en la página 314.

Geometrías replicables y funciones asociadas

Esta sección describe los seis subtipos de geometrías replicables y las funciones que operan sobre ellas. Los subtipos son:

- Puntos
- Líneas
- Polígonos
- Multipuntos
- Multilíneas
- Multipolígonos

La Figura 9 en la página 160 muestra la jerarquía a la que pertenecen estos subtipos y los símbolos visuales asociados a ellos.

Puntos

Un punto es una geometría de dimensión cero que ocupa una sola ubicación en el espacio de coordenadas. Un punto incluye una coordenada X y una coordenada Y que definen esa ubicación. También puede incluir una coordenada Z y una medida.

Un punto es simple y tiene un límite nulo (NULL). Los puntos se suelen utilizar para definir elementos geográficos tales como pozos de petróleo, zonas de interés y elevaciones del terreno.

Funciones que operan sólo sobre el subtipo punto:

PointFromShape
Utiliza como entrada una forma de tipo punto y un identificador de sistema de referencias espaciales, y devuelve un punto. Para obtener más información, consulte la sección "PointFromShape" en la página 219.

ST_Point
Utiliza como entrada una coordenada X, su coordenada Y asociada y el identificador del sistema de referencias espaciales al que pertenecen estas coordenadas y devuelve el punto que definen las coordenadas. Para obtener más información, consulte la sección "ST_Point" en la página 297.

ST_PointFromText
Utiliza como entrada una representación de texto convencional (WKT) OGC correspondiente a un punto y devuelve dicho punto. Para obtener más información, consulte la sección "ST_PointFromText" en la página 298.

ST_PointFromWKB
Utiliza como entrada una representación binaria convencional del tipo

polígono y un identificador de sistema de referencias espaciales, y devuelve un polígono. Para obtener más información, consulte la sección “ST_PointFromWKB” en la página 299.

ST_X Devuelve un valor de coordenada X del tipo de datos ST_Point en forma de número de doble precisión. Para obtener más información, consulte la sección “ST_X” en la página 321.

ST_Y Devuelve un valor de coordenada Y del tipo de datos ST_Point en forma de número de doble precisión. Para obtener más información, consulte la sección “ST_Y” en la página 322.

Z Devuelve un valor de coordenada Z del tipo de datos ST_Point en forma de número de doble precisión. Para obtener más información, consulte la sección “Z” en la página 323.

M Devuelve una medida del tipo de datos ST_Point en forma de número de doble precisión. Para obtener más información, consulte la sección “M” en la página 214.

Líneas

Una línea es un objeto de una sola dimensión que se almacena como secuencia de puntos la cual define una ruta lineal interpolada. La línea es simple si no forma intersección con su interior. Los puntos finales (el límite) de una línea cerrada ocupan el mismo punto en el espacio. Una línea es un anillo si es cerrada y si su interior no forma intersección consigo mismo. Además de las otras propiedades heredades de la geometría de superclase, las líneas tienen longitud. Las líneas se suelen utilizar para definir elementos geográficos lineales, tales como carreteras, ríos y cables de alta tensión.

Una línea simple cuyos punto inicial y punto final coinciden se denomina *anillo*.

Los puntos finales suelen formar el límite de una línea a no ser que esta esté cerrada, en cuyo caso el límite es NULO. El interior de una línea es una ruta conectada que queda dentro de los puntos finales a no ser que esté cerrada, en cuyo caso el interior es continuo.

Funciones que operan sobre líneas:

ST_StartPoint
Utiliza como entrada una línea y devuelve su primer punto. Para obtener más información, consulte la sección “ST_StartPoint” en la página 310.

ST_EndPoint
Utiliza como entrada una línea y devuelve su último punto. Para obtener más información, consulte la sección “ST_Endpoint” en la página 248.

ST_PointN
Utiliza como entrada una línea y un índice al punto *n* y devuelve dicho punto. Para obtener más información, consulte la sección "ST_PointN" en la página 301.

ST_Length
Utiliza como entrada una línea y devuelve su longitud como un número de doble precisión. Para obtener más información, consulte la sección "ST_Length" en la página 278.

ST_NumPoints
Utiliza como entrada una línea y devuelve el número de puntos de su secuencia como un entero. Para obtener más información, consulte la sección "ST_NumPoints" en la página 292.

ST_IsRing
Utiliza como entrada una línea y devuelve un 1 (TRUE) si la línea es un anillo y 0 (FALSE) si no lo es. Para obtener más información, consulte la sección "ST_IsRing" en la página 273.

ST_IsClosed
Utiliza como entrada una línea y devuelve un 1 (TRUE) si la línea está cerrada y 0 (FALSE) si no es así. Para obtener más información, consulte la sección "ST_IsClosed" en la página 269.

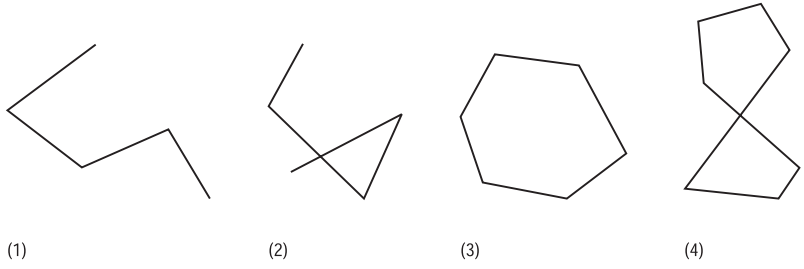


Figura 10. Objetos de línea.

1. Línea simple no cerrada.
2. Línea no simple no cerrada.
3. Línea simple cerrada, por lo tanto anillo.
4. Línea no simple cerrada. No es un anillo.

Polígonos

Un polígono es una superficie de dos dimensiones que se almacena en forma de secuencia de puntos, la cual define el anillo exterior delimitador y los anillos interiores (si los hay) del polígono. Los anillos de un polígono no se pueden solapar. Por lo tanto, por definición, los polígonos son siempre

simples. Los polígonos suelen definir parcelas de terreno, lagos y otros elementos geográficos que tienen una extensión espacial.

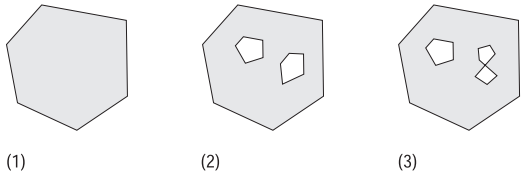


Figura 11. Polígonos.

1. Polígono cuyo límite está definido por un anillo exterior.
2. Polígono cuyo límite está definido por un anillo exterior y dos anillos interiores. La zona situada dentro de los anillos interiores forma parte del exterior del polígono.
3. Polígono válido, pues los anillos forman intersección en un solo punto de tangencia.

El anillo exterior y los anillos interiores (si los hay) definen el límite de un polígono, y el espacio comprendido entre los anillos define el interior del polígono. Los anillos de un polígono pueden tener un punto de tangencia, pero no se pueden cruzar. Además de las otras propiedades heredadas de la geometría de superclase, los polígonos tienen superficie.

Funciones que operan sobre polígonos:

ST_Area
Utiliza como entrada un polígono y devuelve su superficie en forma de número de doble precisión. Para obtener más información, consulte la sección "ST_Area" en la página 225.

ST_ExteriorRing
Utiliza como entrada un polígono y devuelve su anillo exterior en forma de línea. Para obtener más , consulte la sección "ST_ExteriorRing" en la página 252.

ST_NumInteriorRing
Utiliza como entrada un polígono y devuelve el número de anillos interiores que contiene el polígono. Para obtener más información, consulte la sección "ST_NumInteriorRing" en la página 291.

ST_InteriorRingN
Utiliza como entrada un polígono y un índice, y devuelve el anillo interior número *n* en forma de línea. Para obtener más información, consulte la sección "ST_InteriorRingN" en la página 261.

ST_Centroid
Utiliza como entrada un polígono y devuelve un punto que es el centro de la superficie del polígono. Para obtener más información, consulte la sección "ST_Centroid" en la página 233.

ST_PointOnSurface
Utiliza como entrada un polígono y devuelve un punto que con toda seguridad está sobre la superficie del polígono. Para obtener más información, consulte la sección "ST_PointOnSurface" en la página 302.

ST_Perimeter
Utiliza como entrada un polígono y devuelve el perímetro de su superficie. Para obtener más información, consulte la sección "ST_Perimeter" en la página 296.

Multipuntos

Un multipunto es un grupo de puntos y, al igual que sus elementos, tiene una dimensión de 0. Un multipunto es simple si ninguno de sus elementos ocupa el mismo espacio de coordenadas. El límite de un multipunto es nulo (NULL). Los multipuntos se pueden utilizar para definir fenómenos tales como patrones de radiodifusión y casos de un brote epidémico.

Funciones que operan sobre multipuntos:

ST_NumGeometries
Utiliza como entrada un conjunto homogéneo de geometrías y devuelve el número de elementos de geometría base que contiene. Para obtener más información, consulte la sección "ST_NumGeometries" en la página 290.

ST_GeometryN
Utiliza como entrada un conjunto homogéneo de geometrías y un índice, y devuelve la geometría base que ocupa la posición n. Para obtener más información, consulte la sección "ST_GeometryN" en la página 254.

Multilíneas

Una multilínea es un conjunto de líneas. Las multilíneas son simples si forman intersección únicamente en los puntos finales de los elementos de la línea. Las multilíneas no son simples si los interiores de sus elementos forman intersección.

El límite de una multilínea son los puntos finales de sus elementos que no forman intersección. La multilínea está cerrada si todos sus elementos están cerrados. El límite de una multilínea es nulo (NULL) si todos los puntos finales de todos sus elementos forman intersección. Además de las demás propiedades heredadas de la geometría de superclase, las multilíneas tienen longitud. Las multilíneas se utilizan para definir corrientes de agua o redes de carreteras.

Funciones que operan sobre multilíneas:

ST_Length

Utiliza como entrada una multilínea y devuelve la longitud acumulada de todos sus elementos, en forma de número de doble precisión. Para obtener más información, consulte la sección "ST_Length" en la página 278.

ST_IsClosed

Utiliza como entrada una multilínea y devuelve un 1 (TRUE) si la multilínea está cerrada y 0 (FALSE) en caso contrario. Para obtener más información, consulte la sección "ST_IsClosed" en la página 269.

ST_NumGeometries

Utiliza como entrada un conjunto homogéneo y devuelve el número de elementos básicos de geometría que contiene el conjunto. Para obtener más información, consulte la sección "ST_NumGeometries" en la página 290.

ST_GeometryN

Utiliza como entrada un conjunto homogéneo y un índice y devuelve la geometría básica número n. Para obtener más información, consulte la sección "ST_GeometryN" en la página 254.

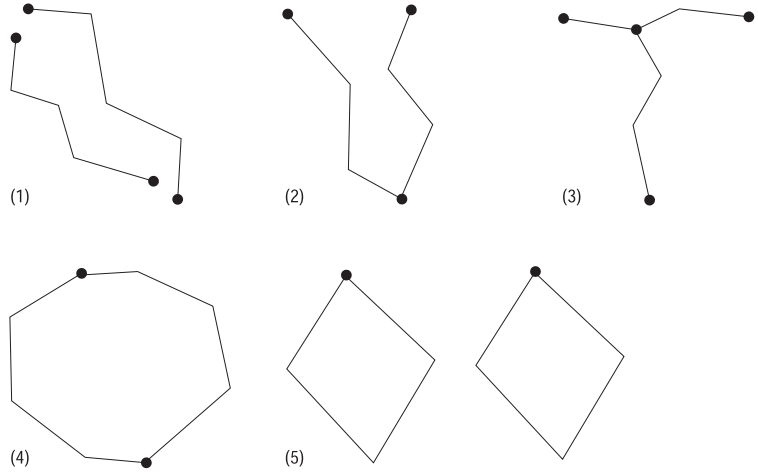
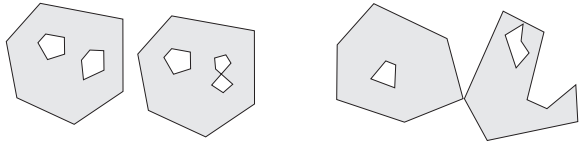


Figura 12. Multilíneas.

1. Multilínea simple cuyo límite está definido por los cuatro puntos finales de sus dos elementos.
2. Multilínea simple, pues sólo se cruzan los puntos finales de los elementos de la multilínea. El límite está definido por los dos puntos finales que no forman intersección.
3. Multilínea no simple, pues el interior de uno de sus elementos forma intersección. El límite de esta multilínea está definido por los cuatro puntos finales, incluido el punto de intersección.
4. Multilínea no cerrada simple. No está cerrada porque sus elementos no están cerrados. Es simple porque ninguno de los interiores de ninguno de sus elementos forman intersección.
5. Multilíneas cerrada simple. Está cerrada porque todos sus elementos están cerrados. Es simple porque ninguno de sus elementos forma intersección en los interiores.

Multipolígonos

El límite de un multipolígono es la longitud acumulada de los anillos exteriores e interiores de sus elementos. El interior de un multipolígono se define como los interiores acumulados de sus polígonos elementales. El límite de los elementos de un multipolígono puede formar intersección únicamente en un punto de tangencia. Además de las otras propiedades que heredan del tipo ST_MultiSurface, los multipolígonos tienen superficie. Los multipolígonos definen elementos geográficos tales como masas forestales o una parcela de terreno no continua, tal como una cadena de islas.



(1) (2)

Figura 13. Multipolígonos.

1. Multipolígono con dos elementos polígonos. El límite está definido por los dos anillos exteriores y los tres anillos interiores.
2. Multipolígono con dos elementos polígonos. El límite está definido por los dos anillos exteriores y los dos anillos interiores. Los dos elementos polígonos forman intersección en un punto de tangencia.

Funciones que operan sobre multipolígonos:

ST_Area
Utiliza como entrada un multipolígono y devuelve la superficie acumulada de sus elementos polígonos, en forma de número de precisión doble. Para obtener más información, consulte la sección "ST_Area" en la página 225.

ST_Centroid
Utiliza como entrada un multipolígono y devuelve un punto que es su centro geométrico ponderado. Para obtener más información, consulte la sección "ST_Centroid" en la página 233.

ST_NumGeometries
Utiliza como entrada un conjunto homogéneo y devuelve el número de elementos de la geometría base que contiene. Para obtener más información, consulte la sección "ST_NumGeometries" en la página 290.

ST_GeometryN
Utiliza como entrada un conjunto homogéneo de geometrías y un índice, y devuelve la geometría base que ocupa la posición n. Para obtener más información, consulte la sección "ST_GeometryN" en la página 254.

Funciones que muestran relaciones y comparaciones, generan geometrías y convierten formatos de valores

Las secciones anteriores contenían una introducción a las tres categorías de funciones espaciales:

- Funciones asociadas a propiedades de las geometrías
- Funciones asociadas a determinadas geometrías

Esta sección contiene una introducción a otras tres categorías:

- Funciones que determinan formas en que los elementos geográficos se pueden relacionar o comparar
- Funciones que generan geometrías nuevas
- Funciones que convierten los valores de una geometría en un formato que se puede importar o exportar

Funciones que muestran relaciones o comparaciones entre elementos geográficos

Varias funciones espaciales devuelven información sobre formas en que los elementos geográficos se pueden relacionar o comparar entre sí. La mayoría de estas funciones devuelven un valor entero. Esta sección proporciona información general sobre las funciones de predicado y luego describe cada función por separado.

Funciones de predicado

Las funciones de predicado devuelven un 1 (TRUE) si una comparación cumple los criterios de la función, o un 0 (FALSE) si la comparación falla. Los predicados que comprueban una relación espacial comparan pares de geometrías que pueden ser de distinto tipo o dimensión.

Los predicados comparan las coordenadas X e Y de las geometrías sometidas. Las coordenadas Z y la medida (si las hay) se pasan por alto. Esto permite comparar geometrías que tienen coordenadas Z o una medida con otras geometrías que no las tienen.

El *Modelo de intersección 9 ampliado dimensionalmente (DE-9IM*) [1]es un enfoque matemático que define la relación espacial entre pares de geometrías de distintos tipos y dimensiones. Este modelo expresa las relaciones espaciales entre todos los tipos de geometrías como intersecciones entre pares de su interior, límite y exterior, teniendo en cuenta la dimensión de las intersecciones resultantes.

Supongamos que tenemos las geometrías *a* y *b*: I(*a*), L(*a*) y E(*a*) representan el interior, el límite y el exterior de *a*, respectivamente. I(b), L(b) y E(b) representan el interior, el límite y el exterior de *b*. La intersección de I(a), L(a) y E(a) con I(b), L(b) y E(b) genera una matriz de 3 por 3. Cada intersección puede dar lugar a geometrías de diferentes dimensiones. Por ejemplo, la

1. El DE-91M fue desarrollado por Clementini y Felice, quienes ampliaron dimensionalmente el Modelo de intersección 9 de Egenhofer y Herring. DE-91M es fruto de la colaboración entre cuatro autores, Clementini, Eliseo, Di Felice y van Osstrom. Publicaron el modelo en "A Small Set of Formal Topological Relationships Suitable for End-User Interaction," D. Abel y B.C. Ooi (Ed.), *Advances in Spatial Database—Third International Symposium. SSD '93*. LNCS 692. Págs. 277-295. El Modelo de intersección 9 de Springer-Verlag Singapore (1993) Egenhofer M.J. y Herring, J., se publicó en "Categorizing binary topological relationships between regions, lines, and points in geographic databases," *Tech. Report, Department of Surveying Engineering*, Universidad de Maine, Orono, ME 1991.

intersección de los límites de dos polígonos consta de un punto y una línea, en cuyo caso la función dim devolvería 1 como dimensión máxima.

La función dim devuelve los valores –1, 0, 1 ó 2. El 1 corresponde al conjunto vacío o dim(null), que se devuelve cuando no se encuentra ninguna intersección.

	Interior	**Límite**	**Exterior**
Interior	dim(I(a) ∩ I(b))	dim(I(a) ∩ L(b))	dim(I(a) ∩ E(b))
Límite	dim(L(a) ∩ I(b))	dim(L(a) ∩ L(b))	dim(L(a) ∩ E(b))
Exterior	dim(E(a) ∩ I(b))	dim(E(a) ∩ L(b))	dim(E(a) ∩ E(b))

Los resultados de los predicados de relación espacial se pueden entender o comprobar comparando los resultados del predicado con una matriz patrón que representa los valores aceptables para el DE-9IM.

La matriz patrón contiene los valores aceptables para cada una de las celdas de la matriz de intersecciones. Los valores patrón posibles son:

T Debe existir una intersección, dim = 0, 1 ó 2.

F No debe existir intersección, dim = -1.

***** No importa si existe o no intersección, dim = -1, 0, 1 ó 2.

0 Debe existir una intersección y su dimensión máxima debe ser 0, dim = 0.

1 Debe existir una intersección y su dimensión máxima debe ser 1, dim = 1.

2 Debe existir una intersección y su dimensión máxima debe ser 2, dim = 2.

Por ejemplo, la siguiente matriz patrón correspondiente al predicado ST_Within incluye los valores T, F y *.

Tabla 41. Matriz correspondiente a ST_Within. Matriz patrón del predicado ST_Within correspondiente a combinaciones de geometrías.

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	T	*	F
	Límite	*	*	F
	Exterior	*	*	*

El predicado ST_Within devuelve TRUE si los interiores de ambas geometrías forman intersección y si el interior y el límite de *a* no forma intersección con el exterior de *b.* Las demás condiciones no importan.

Cada predicado tiene al menos una matriz patrón, pero algunos necesitan más de una matriz para describir las relaciones de diversas combinaciones de tipos de geometrías.

ST_Equals

ST_Equals devuelve un 1 (TRUE) si las dos geometrías del mismo tipo tienen valores de coordenadas X,Y idénticos.

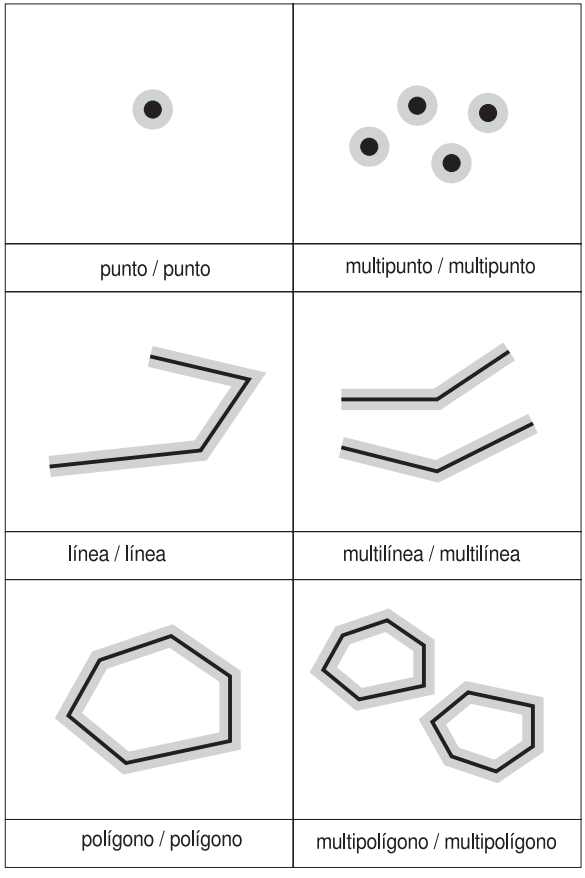


Figura 14. ST_Equals. Las geometrías son iguales si tienen coordenadas X,Y coincidentes.

Tabla 42. Matriz correspondiente a igualdad. La matriz de patrón DE-9IM correspondiente a igualdad asegura que los interiores forman intersección y que ninguna parte del interior o del límite de ninguna geometría forma intersección con el exterior de la otra.

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	T	*	F
	Límite	*	*	F
	Exterior	F	F	*

Para obtener más información, consulte la sección "ST_Equals" en la página 251.

ST_OrderingEquals

ST_OrderingEquals compara dos geometrías y devuelve un 1 (TRUE) si las geometrías son iguales y las coordenadas están en el mismo orden; de lo contrario, devuelve un 0 (FALSE). Para obtener más información, consulte la sección "ST_OrderingEquals" en la página 293.

ST_Disjoint

ST_Disjoint devuelve un 1 (TRUE) si la intersección de las dos geometrías es un conjunto vacío.

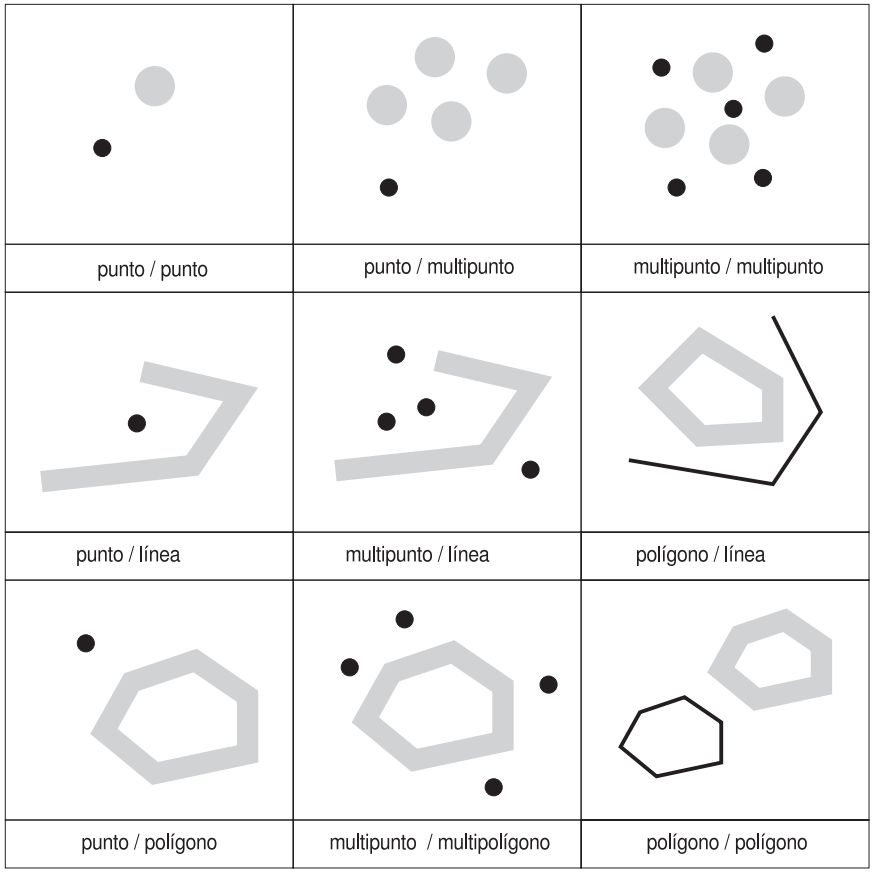


Figura 15. ST_Disjoint. Las geometrías están desunidas si no forman intersección entre sí en ninguna forma.

Tabla 43. Matriz correspondiente a ST_Disjoint. La matriz patrón del predicado ST_Disjoint indica que ni los interiores ni los límites de ninguna geometría forman intersección.

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	F	F	*
	Límite	F	F	*
	Exterior	*	*	*

Para obtener más información, consulte la sección “ST_Disjoint” en la página 245.

ST_Intersects

ST_Intersects devuelve un 1 (TRUE) si la intersección no da como resultado un conjunto vacío. La intersección devuelve exactamente el resultado opuesto a ST_Disjoint.

El predicado ST_Intersects devuelve TRUE si las condiciones de algunas de las siguientes matrices patrón devuelven TRUE.

Tabla 44. Matriz correspondiente a ST_Intersects (1). El predicado ST_Intersects devuelve TRUE si los interiores de ambas geometrías forman intersección.

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	T	*	*
	Límite	*	*	*
	Exterior	*	*	*

Tabla 45. Matriz correspondiente a ST_Intersects (2). El predicado ST_Intersects devuelve TRUE si el límite de la primera geometría forma intersección con el límite de la segunda.

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	*	T	*
	Límite	*	*	*
	Exterior	*	*	*

Tabla 46. Matriz correspondiente a ST_Intersects (3). El predicado ST_Intersects devuelve TRUE si el límite de la primera geometría forma intersección con el interior de la segunda.

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	*	*	*
	Límite	T	*	*
	Exterior	*	*	*

Tabla 47. Matriz correspondiente a ST_Intersects (4). El predicado ST_Intersects devuelve TRUE si los límites de alguna de las geometrías forman intersección.

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	*	*	*
	Límite	*	T	*
	Exterior	*	*	*

Para obtener más información, consulte la sección “ST_Intersects” en la página 268.

EnvelopesIntersect
Esta función devuelve un 1 (TRUE) su las envolventes de dos geometrías forman intersección. Es una función muy útil que implanta de forma eficiente ST_Intersects (`ST_Envelope(g1),ST_Envelope(g2)`). Para obtener más información, consulte la sección "EnvelopesIntersect" en la página 203.

ST_Touches
ST_Touches devuelve un 1 (TRUE) si ninguno de los puntos comunes a ambas geometrías forman intersección con los interiores de ambas geometrías. Al menos una geometría debe ser una línea, un polígono, una multilínea o un multipolígono.

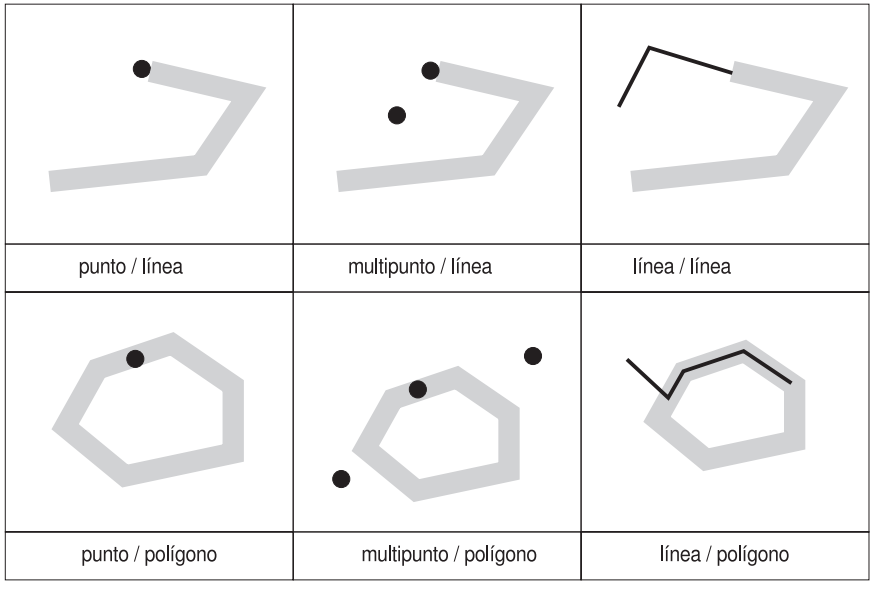


Figura 16. ST_Touches

Las matrices patrón muestran que el predicado ST_Touches devuelve TRUE cuando los interiores de las geometrías no forman intersección y el límite de alguna de las geometrías forma intersección con el interior o el límite de la otra.

Tabla 48. Matriz correspondiente a ST_Touches (1)

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	F	T	*
	Límite	*	*	*
	Exterior	*	*	*

Tabla 49. Matriz correspondiente a ST_Touches (2)

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	F	*	*
	Límite	T	*	*
	Exterior	*	*	*

Tabla 50. Matriz correspondiente a ST_Touches (3)

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	F	*	*
	Límite	*	T	*
	Exterior	*	*	*

Para obtener más información, consulte la sección "ST_Touches" en la página 313.

ST_Overlaps

ST_Overlaps compara dos geometrías de la misma dimensión. Devuelve un 1 (TRUE) si el resultado de su intersección da lugar a una geometría distinta de ambas, pero que tiene la misma dimensión.

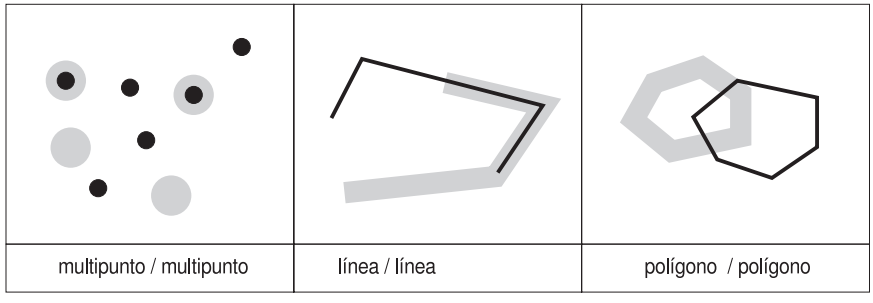


Figura 17. ST_Overlaps

La matriz patrón de la Tabla 51 en la página 183 se aplica a los solapamientos polígono/polígono, multipunto/multipunto y multipolígono/multipolígono. Para estas combinaciones, el predicado de solapamiento devuelve TRUE si el interior de ambas geometrías forma intersección con el interior y el exterior de la otra.

Tabla 51. Matriz correspondiente a ST_Overlaps (1)

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	T	*	T
	Límite	*	*	*
	Exterior	T	*	*

La matriz patrón de la Tabla 52 se aplica a solapamientos línea/línea y multilínea/multilínea. En este caso, la intersección de las geometrías debe dar lugar a una geometría con una dimensión de 1 (otra línea). Si la dimensión de la intersección de los interiores es 1, el predicado ST_Overlaps devolvería FALSE, pero el predicado ST_Crosses devolvería TRUE.

Tabla 52. Matriz correspondiente a ST_Overlaps (2)

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	1	*	T
	Límite	*	*	*
	Exterior	T	*	*

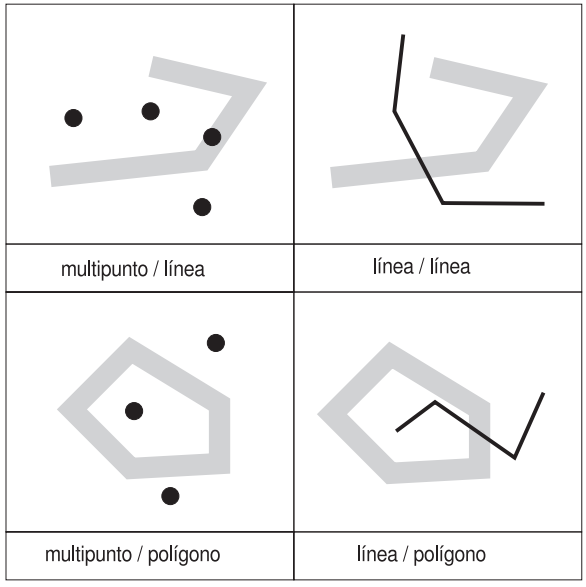
Para obtener más información, consulte la sección “ST_Overlaps” en la página 294.

ST_Crosses

ST_Crosses utiliza como entrada dos geometrías y devuelve un 1 1 (TRUE) si:

- La intersección da lugar a una geometría cuya dimensión es menor que la dimensión máxima de las geometrías de origen.
- El conjunto resultante de la intersección es interior respecto a las dos geometrías de origen.

ST_Crosses devuelve un 1 (TRUE) sólo para las comparaciones multipunto/polígono, multipunto/línea, línea/línea, línea/polígono y línea/multipolígono.



La matriz patrón de la Tabla 53 es aplicable a las combinaciones multipunto/línea, multipunto/multilínea, multipunto/polígono, multipunto/multipolígono, línea/polígono y línea/multipolígono. La matriz indica que los interiores deben formar intersección y que el interior de la geometría principal (geometría *a*) debe formar intersección con el exterior de la geometría secundaria (geometría *b*).

Tabla 53. Matriz correspondiente a ST_Crosses (1)

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	T	*	T
	Límite	*	*	*
	Exterior	*	*	*

La matriz patrón de la Tabla 54 en la página 185 se aplica a línea/línea, línea/multilínea y multilínea/multilínea. La matriz indica que la dimensión de la intersección de los interiores debe ser 0 (intersección en un punto). Si la dimensión de esta intersección es 1 (intersección en una línea), el predicado ST_Crosses devuelve FALSE; sin embargo, el predicado ST_Overlaps devuelve TRUE.

Tabla 54. Matriz correspondiente a ST_Crosses (2)

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	0	*	*
	Límite	*	*	*
	Exterior	*	*	*

Para obtener más información, consulte la sección "ST_Crosses" en la página 240.

ST_Within

ST_Within devuelve un 1 (TRUE) si la primera geometría queda completamente dentro de la segunda geometría. ST_Within devuelve exactamente el resultado opuesto a ST_Contains.

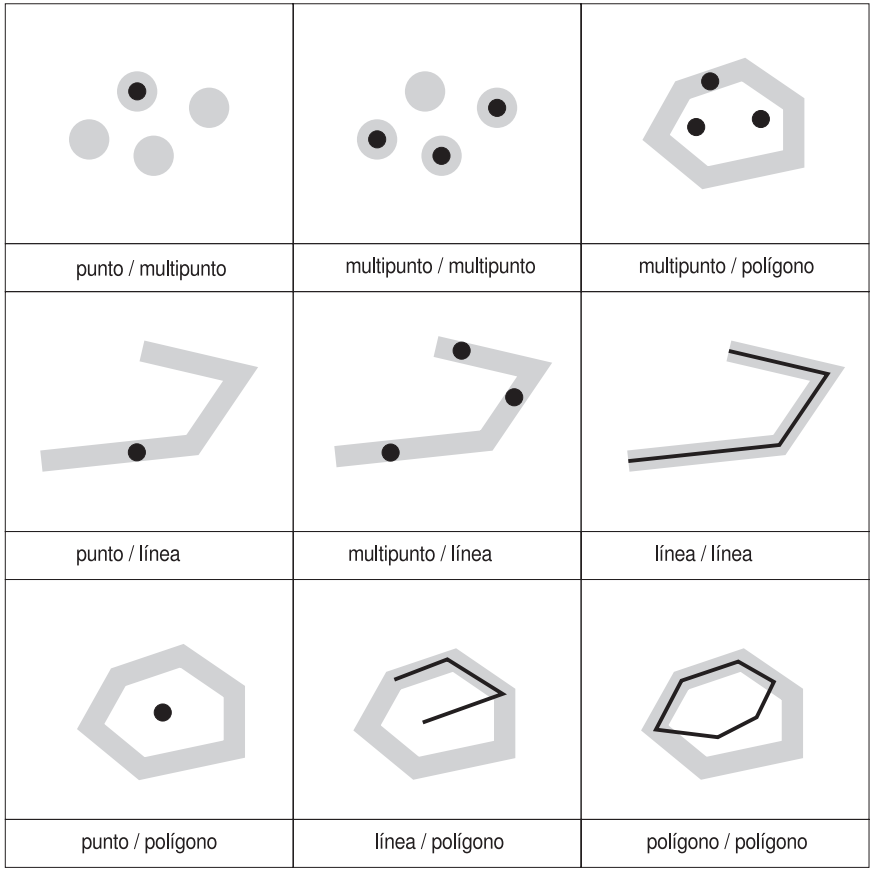


Figura 18. ST_Within

La matriz patrón del predicado ST_Within indica que los interiores de ambas geometrías deben formar intersección, y que el interior y el límite de la geometría principal (geometría *a*) no deben formar intersección con el exterior de la secundaria (geometría *b*).

Tabla 55. Matriz correspondiente a ST_Within

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	T	*	F
	Límite	*	*	F
	Exterior	*	*	*

Para obtener más información, consulte la sección “ST_Within” en la página 317.

ST_Contains

ST_Contains devuelve un 1 (TRUE) si la segunda geometría queda completamente contenida en la primera geometría. El predicado ST_Contains devuelve exactamente el resultado opuesto al predicado ST_Within.

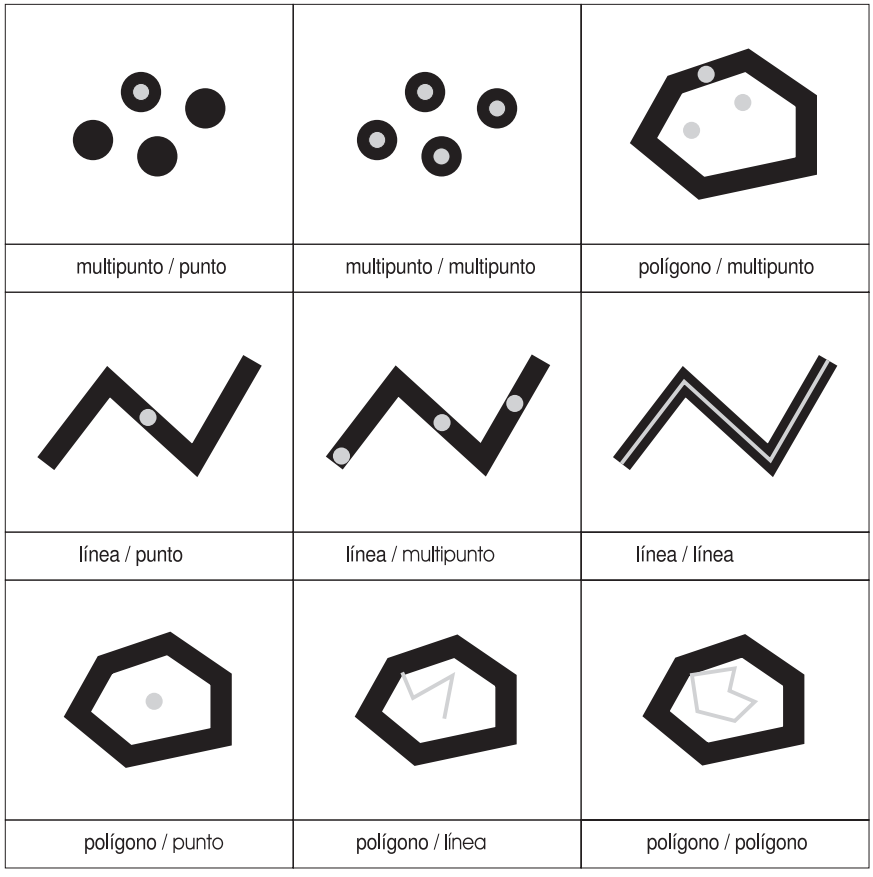


Figura 19. ST_Contains

La matriz patrón del predicado ST_Contains indica que los interiores de ambas geometrías deben forman intersección y que el interior y el límite de la geometría secundaria (geometría *b*) no debe formar intersección con el exterior de la principal (geometría *a*).

Tabla 56. Matriz correspondiente a ST_Contains

		b		
		Interior	**Límite**	**Exterior**
a	**Interior**	T	*	*
	Límite	*	*	*
	Exterior	F	F	*

Para obtener más información, consulte la sección “ST_Contains” en la página 234.

ST_Relate

La función ST_Relate compara dos geometrías y devuelve un 1 (TRUE) si las geometrías cumplen con las condiciones especificadas por la serie de la matriz patrón DE-91M; si no es así, la función devuelve un 0 (FALSE). Para obtener más información, consulte la sección "ST_Relate" en la página 307.

ST_Distance

La función ST_Distance indica la distancia mínima que separa dos elementos geográficos separados. Si los elementos no están separados, la función indicará una distancia mínima de 0.

Por ejemplo, ST_Distance podría indicar la distancia más corta que debe recorrer un avión entre dos lugares. La Figura 20 ilustra esta información.

Figura 20. Distancia mínima entre dos ciudades. ST_Distance puede tomar las coordenadas correspondientes a las ciudades de Los Ángeles y Chicago como entrada y devolver un valor que indique la distancia mínima entre estas ubicaciones.

Para obtener más información, consulte la sección "ST_Distance" en la página 247.

Funciones que generan geometrías nuevas a partir de geometrías existentes

Spatial Extender ofrece predicados y funciones de transformación que generan geometrías nuevas a partir de geometrías existentes.

ST_Intersection

La función ST_Intersection devuelve el conjunto de intersecciones de dos geometrías. El conjunto de intersecciones siempre se devuelve como un grupo que es la dimensión mínima de las geometrías de origen. Por ejemplo, para una línea que forma intersección con un polígono, la función de intersección devuelve una multilínea formada por la parte de la línea que es común al interior y al límite del polígono. La multilínea contiene más de una línea si la línea de origen forma intersección con el polígono en dos o más segmentos

discontinuos. Si las geometrías no forman intersección o si la intersección da como resultado una dimensión inferior a las de las geometrías de origen, se devuelve una geometría vacía.

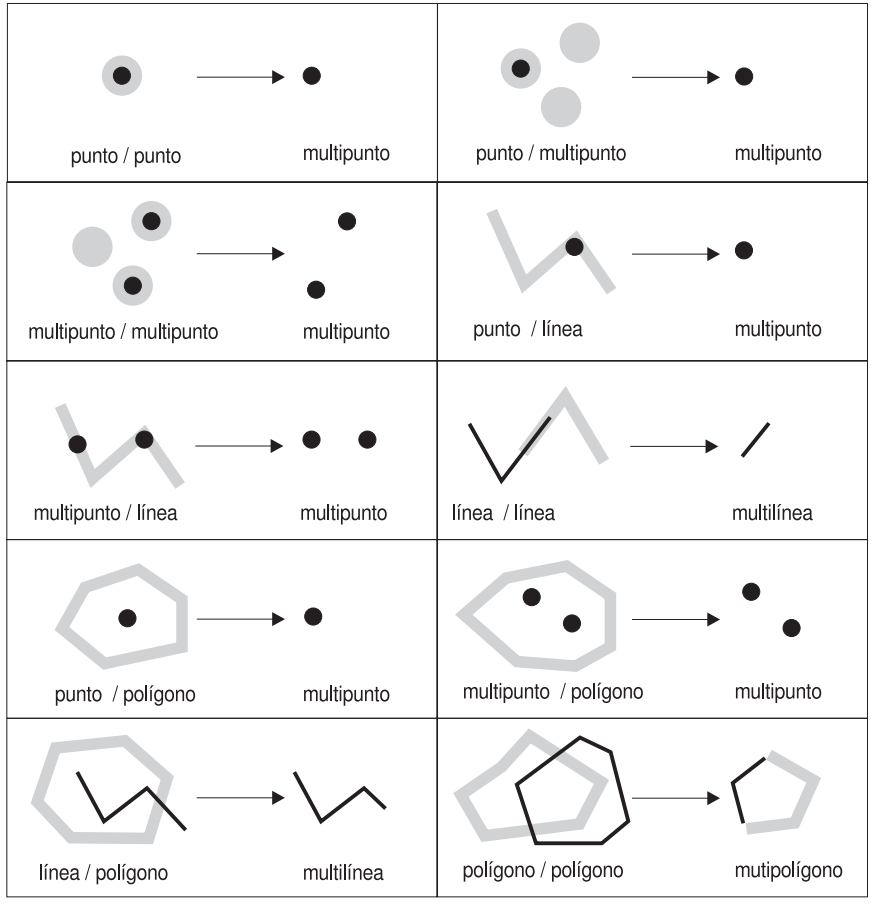


Figura 21. ST_Intersection. Ejemplos de la función ST_Intersection.

Para obtener más información, consulte la sección “ST_Intersection” en la página 266.

ST_Difference

ST_Difference utiliza como entrada dos geometrías. La primera se denomina *geometría primaria*; la segunda se denomina *geometría secundaria*. La función ST_Difference devuelve la parte de la geometría principal que no forma intersección con la geometría secundaria. Se trata del AND NOT lógico de espacio. La función ST_Difference sólo funciona sobre geometrías de la misma dimensión y devuelve un grupo que tiene la misma dimensión que las geometrías de origen. En el caso de que las geometrías de origen fueran

iguales, se devolvería una geometría vacía. Si las dimensiones de la geometrías de entrada de la función ST_Difference son diferentes, ST_Difference devuelve un valor nulo.

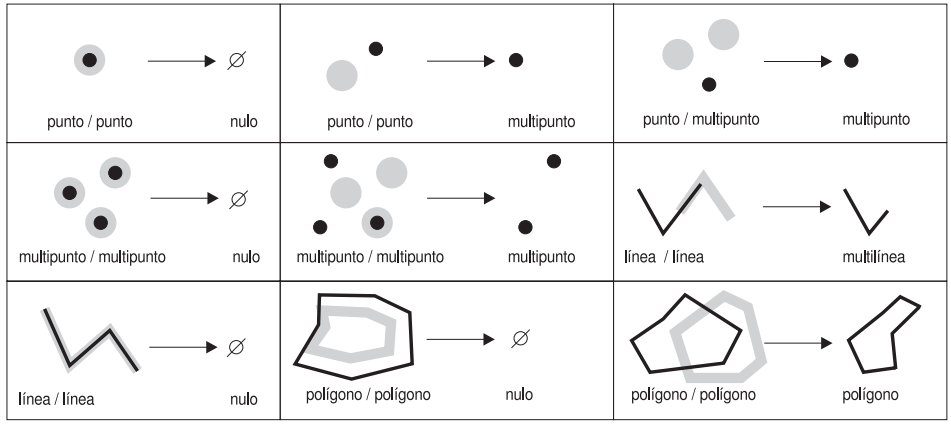


Figura 22. ST_Difference

Para obtener más información, consulte la sección "ST_Difference" en la página 242.

ST_Union

La función ST_Union devuelve el conjunto de unión de dos geometrías. Es el OR lógico de espacio. Las geometrías de origen deben tener la misma dimensión. ST_Union siempre devuelve el resultado como un grupo.

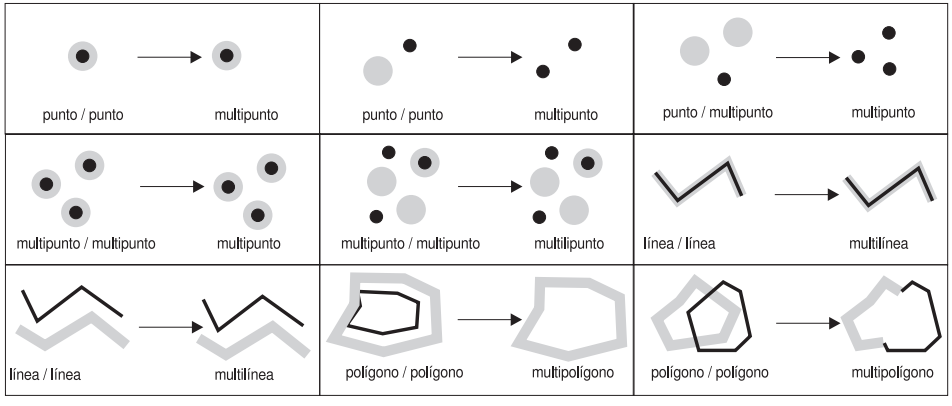


Figura 23. ST_Union

Para obtener más información, consulte la sección "ST_Union" en la página 316.

ST_SymmetricDiff

La función ST_SymmetricDiff devuelve la diferencia simétrica (el operador lógico XOR para el espacio) de dos geometrías que forman intersección y que tienen la misma dimensión. Si estas geometrías son iguales, ST_SymmetricDiff devuelve una geometría vacía. Si las geometrías no son iguales, una porción de una o ambas de ellas estará fuera de la zona de intersección. ST_SymmetricDiff devuelve la porción o porciones de no intersección en forma de conjunto de geometrías; por ejemplo, en forma de multipolígono.

Si las geometrías de entrada de ST_SymmetricDiff tienen dimensiones diferentes, la función devuelve un valor nulo.

ST_Buffer

La función ST_Buffer genera una geometría rodeando una geometría a una distancia especificada. Da como resultado un polígono cuando se rodea una geometría principal o cuando los elementos de un grupo están lo suficientemente juntos para que todos los polígonos rodeados se solapan. Sin embargo, si hay suficiente separación entre los elementos de un conjunto rodeado, se obtendrá como resultado polígonos individuales rodeados; en este caso la función ST_Buffer devuelve un multipolígono.

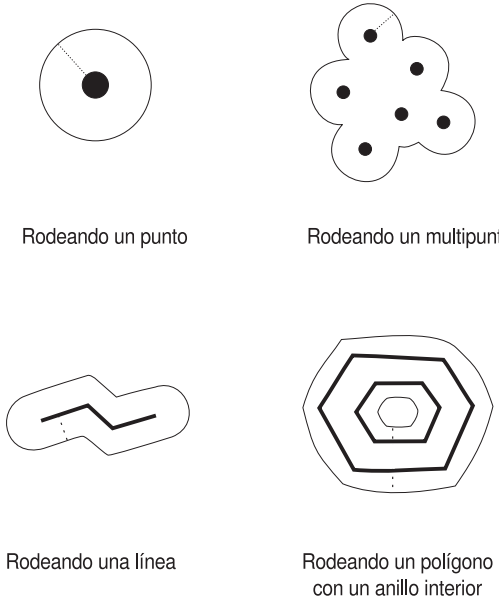


Figura 24. ST_Buffer

La función ST_Buffer acepta distancias positivas y negativas; sin embargo, sólo las geometrías con una dimensión de dos (polígonos y multipolígonos) aplican una distancia de rodeo negativa. El valor absoluto de la distancia de

rodeo se utiliza cuando la dimensión de la geometría de origen es menor que 2 (todas las geometrías que no son polígono ni multipolígono).

En general, para anillos exteriores, las distancias de rodeo positivas generan anillos de polígono que están lejos del centro de la geometría de origen; las distancias de rodeo negativas generan anillos de polígono o multipolígono hacia el centro. Para anillos interiores de un polígono o multipolígono, una distancia de rodeo positiva genera un anillo de rodeo hacia el centro y una distancia de rodeo negativa genera un anillo de rodeo alejado del centro.

El proceso de rodeo fusiona polígonos que se solapan. Las distancias negativas superiores a la mitad de la anchura interior máxima de un polígono dan como resultado una geometría vacía.

Para obtener más información, consulte la sección "ST_Buffer" en la página 231.

LocateAlong

Para geometrías que tienen medidas, la ubicación de una determinada medida se puede encontrar con la función LocateAlong. LocateAlong devuelve la ubicación en forma de multipunto. Si la dimensión de la geometría de origen es 0 (por ejemplo, un punto y un multipunto), es necesaria una coincidencia exacta, y los puntos que tienen un valor de medida coincidente se devuelven en forma de multipunto. Sin embargo, si la dimensión de las geometrías de origen es mayor que 0, la ubicación se interpola. Por ejemplo, si el valor de medida entrado es 5,5 y las medidas de los vértices de una línea son 3, 4, 5, 6 y 7 respectivamente, se devuelve el punto interpolado que está exactamente a mitad de camino entre los vértices cuyas medidas son 5 y 6.

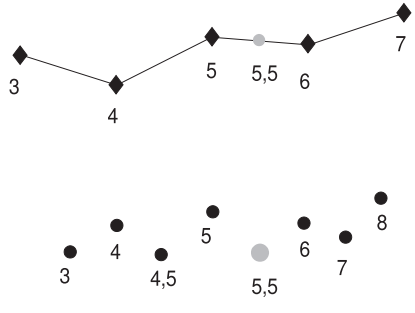


Figura 25. LocateAlong

Para obtener más información, consulte la sección "LocateAlong" en la página 210.

LocateBetween

La función LocateBetween devuelve una serie de rutas o ubicaciones que quedan entre dos valores de medida de una geometría de origen que tiene medidas. Si la dimensión de la geometría de origen es 0, LocateBetween devuelve un multipunto que contiene todos los puntos cuyas medidas quedan entre las dos medidas originales. Para las geometrías de origen cuya dimensión es mayor que 0, LocateBetween devuelve una multilínea si se puede interpolar una ruta; de lo contrario, LocateBetween devuelve un multipunto que contiene las ubicaciones de los puntos. Se devuelve un punto vacío si LocateBetween no puede interpolar una ruta ni encontrar ninguna ubicación entre las medidas. LocateBetween realiza un examen inclusivo de las geometrías; por lo tanto, las medidas de las geometrías deben ser mayores o iguales que la medida de *origen* y menores o iguales que la medida de *destino*.

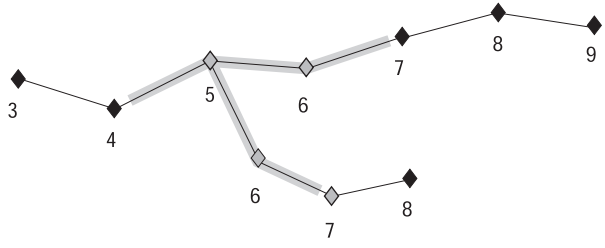


Figura 26. LocateBetween

Para obtener más información, consulte la sección “LocateBetween” en la página 212.

ST_ConvexHull

La función ST_ConvexHull devuelve el polígono de casco convexo de cualquier geometría que tenga al menos tres vértices que formen una forma convexa. Si los vértices de la geometría no forman una forma convexa, ST_ConvexHull devuelve un nulo. ST_ConvexHull suele ser el primer paso del proceso utilizado para crear una red TIN a partir de una serie de puntos.

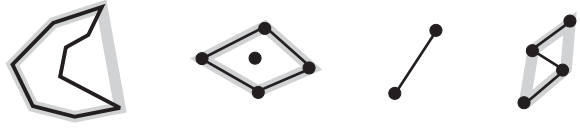


Figura 27. ST_ConvexHull

Para obtener más información, consulte la sección “ST_ConvexHull” en la página 236.

ST_Polygon
Genera un polígono a partir de una línea. Para obtener más información, consulte la sección “ST_Polygon” en la página 306.

Funciones que convierten el formato de los valores de una geometría

Spatial Extender da soporte a tres formatos de intercambio de datos GIS:

- Representación de texto convencional
- Representación binaria convencional
- Representación de forma ESRI

Representación de texto convencional

Spatial Extender tiene varias funciones que generan geometrías a partir de descripciones de texto.

ST_WKTToSQL
Crea una geometría a partir de una representación de texto de cualquier tipo de geometría. No se debe especificar ningún identificador de sistema de referencias espaciales. Para obtener más información, consulte la sección “ST_WKTToSQL” en la página 320.

ST_GeomFromText
Crea una geometría a partir de una representación de texto de cualquier tipo de geometría. Se debe especificar un identificador de sistema de referencias espaciales. Para obtener más información, consulte la sección “ST_GeomFromText” en la página 257.

ST_PointFromText
Crea un punto a partir de una representación de texto de un punto. Para obtener más información, consulte la sección “ST_PointFromText” en la página 298.

ST_LineFromText
Crea una línea a partir de una representación de texto de una línea. Para obtener más información, consulte la sección “ST_LineFromText” en la página 280.

ST_PolyFromText
Crea un polígono a partir de una representación de texto de un polígono. Para obtener más información, consulte la sección “ST_PolyFromText” en la página 303.

ST_MPointFromText
Crea un multipunto a partir de una representación de multipunto. Para obtener más información, consulte la sección “ST_MPointFromText” en la página 286.

ST_MLineFromText
Crea una multilínea a partir de una representación de multilínea. Para obtener más información, consulte la sección "ST_MLineFromText" en la página 283.

ST_MPolyFromText
Crea un multipolígono a partir de una representación de multipolígono. Para obtener más información, consulte la sección "ST_MPolyFromText" en la página 288.

La representación de texto es una serie de caracteres ASCII. Permite intercambiar una geometría en formato de texto ASCII. Estas funciones no necesitan la definición de ninguna estructura de programa especial para correlacionar una representación binaria. Por lo tanto, se pueden utilizar en un programa 3GL o 4GL.

La función ST_AsText convierte un valor de geometría existente en una representación de texto. Para obtener más información, consulte la sección "ST_AsText" en la página 228.

Para ver una descripción detallada de las representaciones de texto convencionales, consulte la sección "Representaciones de texto convencionales de OGC" en la página 337.

Representaciones binarias convencionales

Spatial Extender tiene varias funciones que generan geometrías a partir de representaciones binarias convencionales (WKB).

ST_WKBToSQL
Crea una geometría a partir de una representación binaria convencional de cualquier tipo de geometría. No se debe especificar ningún identificador de sistema de referencias espaciales. Para obtener más información, consulte la sección "ST_WKBToSQL" en la página 318.

ST_GeomFromWKB
Crea una geometría a partir de una representación binaria convencional de cualquier tipo de geometría. Se debe especificar un identificador de sistema de referencias espaciales. Para obtener más información, consulte la sección "ST_GeomFromWKB" en la página 259.

ST_PointFromWKB
Crea un punto a partir de una representación binaria convencional de un punto. Para obtener más información, consulte la sección "ST_PointFromWKB" en la página 299.

ST_LineFromWKB
Crea una línea a partir de una representación binaria convencional de

una línea. Para obtener más información, consulte la sección "ST_LineFromWKB" en la página 281.

ST_PolyFromWKB
Crea un polígono a partir de una representación binaria convencional de un polígono. Para obtener más información, consulte la sección "ST_PolyFromWKB" en la página 304.

ST_MPointFromWKB
Crea un multipunto a partir de una representación binaria convencional de un multipunto. Para obtener más información, consulte la sección "ST_MPointFromWKB" en la página 287.

ST_MLineFromWKB
Crea una multilínea a partir de una representación binaria convencional de una multilínea. Para obtener más información, consulte la sección "ST_MLineFromWKB" en la página 284.

ST_MPolyFromWKB
Crea un multipolígono a partir de una representación binaria convencional de un multipolígono. Para obtener más información, consulte la sección "ST_MPolyFromWKB" en la página 289.

La representación binaria convencional es una corriente continua de bytes. Permite intercambiar una geometría entre un cliente ODBC y una base de datos SQL en formato binario. Estas funciones geométricas necesitan la definición de estructuras C para correlacionar la representación binaria. Por lo tanto, están destinadas a ser utilizadas dentro de un programa 3GL y no se ajustan a un entorno 4GL.

La función ST_AsBinary convierte un valor de geometría existente en una representación binaria convencional. Para obtener más información, consulte la sección "ST_AsBinary" en la página 227.

Para ver una descripción detallada de las representaciones binarias convencionales, consulte la sección "Representaciones binarias convencionales (representaciones WKB) de OGC" en la página 342.

Representación de forma ESRI

Spatial Extender tiene varias funciones que generan geometrías a partir de una representación de forma ESRI. La representación de forma ESRI da soporte a coordenadas Z y a medidas, además de las representaciones de dos dimensiones soportadas por las representaciones binarias convencionales y las representaciones de texto.

ShapeToSQL
Crea una geometría a partir de una forma de cualquier tipo de geometría. No se debe especificar ningún identificador de sistema de

referencias espaciales. Para obtener más información, consulte la sección “ShapeToSQL” en la página 223.

GeometryFromShape
Crea una geometría a partir de una forma de cualquier tipo de geometría. Se debe especificar un identificador de sistema de referencias espaciales. Para obtener más información, consulte la sección “GeometryFromShape” en la página 205.

PointFromShape
Crea un punto a partir de una forma de punto. Para obtener más información, consulte la sección “PointFromShape” en la página 219.

LineFromShape
Crea una línea a partir de una forma de polilínea. Para obtener más información, consulte la sección “LineFromShape” en la página 208.

PolyFromShape
Crea un polígono a partir de una forma de polilínea. Para obtener más información, consulte la sección “PolyFromShape” en la página 221.

MPointFromShape
Crea un multipunto a partir de una forma de multipunto. Para obtener más información, consulte la sección “MPointFromShape” en la página 217.

MLineFromShape
Crea un multilínea a partir de una forma de polilínea que consta de varias partes. Para obtener más información, consulte la sección “MLine FromShape” en la página 215.

MPolyFromShape
Crea un multipolígono a partir de una forma de polígono que consta de varias partes. Para obtener más información, consulte la sección “MPolyFromShape” en la página 218.

La sintaxis general de estas funciones es la misma. El primer argumento es la representación de forma que se entra como un tipo de datos BLOB. El segundo argumento es el identificador de referencias espaciales que se asignará a la geometría. Por ejemplo, la función GeometryFromShape tiene la siguiente sintaxis:

```
GeometryFromShape(shapegeometry, SRID)
```

Para correlacionar la representación binaria, estas funciones de forma necesitan la definición de estructuras C. Por lo tanto, están destinadas a ser utilizadas dentro de un programa 3GL y no se ajustan a un entorno 4GL.

La función AsBinary convierte un valor de geometría en una representación de forma ESRI. Para obtener más información, consulte la sección “AsShape” en la página 202.

Para ver una descripción detallada de las representaciones de forma, consulte la sección “Las representaciones de forma ESRI” en la página 346.

Capítulo 14. Funciones espaciales para consultas SQL

Este capítulo contiene las funciones disponibles que sirven para consultar datos espaciales. Cada función se describe en una sección que le muestra la sintaxis, el tipo de datos devuelto y ejemplos de código. Algunos de los ejemplos de este capítulo incluyen una sentencia CREATE TABLE en la que hay una o más columnas definidas como columnas espaciales.

Tal como se indicó en la sección ″Acerca de los tipos de datos espaciales″ del Capítulo 4 (″Definir columnas espaciales, registrarlas como capas y habilitar un geocodificador para mantenerlas″), los tipos de datos espaciales forman una jerarquía, en la que ST_Geometry es la raíz. Cuando en el presente manual se indique que un valor de un tipo subordinado de esta jerarquía se puede utilizar como entrada para una función, se da por supuesto que la función también puede aceptar como entrada un valor de cualquier tipo que esté por debajo en la jerarquía.

Por ejemplo, en el Capítulo 14, la sección ″Is3d″ indica que la función Is3d utiliza como entrada un valor cuyo tipo de datos es ST_Geometry. Esto significa que, como alternativa, la función Is3d puede aceptar como entrada un valor perteneciente a cualquier tipo de datos soportado que sea un subordinado de ST_Geometry: ST_Point, ST_Curve, ST_LineString, etc. Para ver un esquema de los tipos de datos subordinados, vea la Figura 6 del manual *Guía y consulta del usuario*.

Las siguientes consideraciones son aplicables a las funciones espaciales:

- Los ejemplos de este capítulo están calificados con el nombre de biblioteca db2gse. En lugar de calificar explícitamente con db2gse cada función espacial y tipo de datos, puede incluir db2gse en la vía de acceso de la función.
- Si ST_Union utiliza como entrada dos puntos que tienen las mismas coordenadas, la función devuelve un solo punto. Si este punto se proporciona como entrada a ST_IsSimple, esta función devuelve un 1 (TRUE; es decir, sí, el punto es simple).
- Si una función espacial recibe un nulo como parámetro de entrada, la función devuelve un nulo como parámetro de salida.
- Antes de insertar datos en una columna espacial:
  - Es posible que tenga que aumentar el parámetro udf_mem_sz. El valor recomendado es 2048. Si 2048 no resulta adecuado, aumente el parámetro udf_mem_sz en incrementos de 256.

– Debe registrarlo como una capa. Para obtener más información sobre cómo registrar una columna espacial como una capa, consulte "Capítulo 4. Cómo definir columnas espaciales, registrarlas como capas y habilitar un geocodificador para que las mantenga" en la página 45.

Anidamiento

El anidar juntas funciones de geometría puede originar problemas si la función anidada interna devuelve un tipo de geometría que no es compatible con la función solicitante externa. Este problema se puede resolver si el tipo de geometría de la función anidada interna se puede convertir a un tipo que sea aceptable para la función solicitante externa.

Conversión de ST_Geometry a un subtipo

El tipo de geometría de las funciones que devuelven el supertipo ST_Geometry se puede convertir a un subtipo utilizando la función Treat. Por ejemplo, la función ST_Union devuelve valores de ST_Geometry. Cuando la función ST_Union se anida dentro de la función ST_PointOnSurface, ST_PointOnSurface devuelve el error siguiente:

```
SQL00440N No se ha encontrado ninguna función llamada "ST_POINTONSURFACE"
en la vía de acceso de la función que tenga argumentos compatibles.
SQLSTATE=42884
```

La función ST_PointOnSurface utiliza como entrada los tipos de geometría ST_Polygon o ST_MultiPolygon, pero no el tipo ST_Geometry devuelto por ST_Union, aunque el valor devuelto por la función ST_Union sea ST_MultiPolygon. Por tanto, en este caso, es necesario convertir el tipo de geometría de la función ST_Union a ST_MultiPolygon.

Por ejemplo, si la tabla COUNTIES se une consigo mismo mediante la unión de su columna de polígono COUNTY, la función Treat se tendrá que aplicar al resultado de la función ST_Union para convertir el tipo ST_Geometry a ST_MultiPolygon antes de poder aplicar la función ST_PointOnSurface.

```
SELECT ST_Astext(ST_PointOnSurface(
          TREAT ( ST_Union(c1.county, c2.county) AS ST_MultiPolygon)))
FROM counties AS c1, counties AS c2;
```

Si la función ST_Union devuelve un valor de ST_MultiPolygon, la función Treat lo convierte al tipo de datos ST_MultiPolygon. Si la función ST_Union no devuelve un valor de ST_MultiPolygon, la función Treat devuelve un error de ejecución.

Para obtener más información sobre el tratamiento de subtipos, vea el manual *Consulta de SQL*.

Conversión de un conjunto de geometrías en una geometría básica

La función ST_GeometryN convierte un elemento de un conjunto de geometrías en una geometría básica, que puede ser luego utilizada por la función solicitante externa.

Por ejemplo, el valor devuelto por la función ST_Union es siempre un conjunto de geometrías devuelto en forma de ST_Geometry. Utilice la función Treat para convertir el tipo ST_Geometry en un subtipo, el cual puede ser: ST_MultiPoint, ST_MultiLineString, ST_MultiPolygon, ST_GeomCollection, ST_MultiCurve o ST_MultiSurface. Aplique la función ST_GeometryN al resultado de convertir el conjunto de geometrías en una geometría básica por medio de la función Treat.

Por ejemplo, para aplicar la función ST_ExteriorRing al resultado de la función ST_Union, del ejemplo de la sección anterior, utilice primero la función ST_GeometryN para extraer un elemento de polígono.

```
SELECT ST_AsText(ST_ExteriorRing(ST_GeometryN(
          TREAT ( ST_Union(c1.county, c2.county) AS ST_MultiPolygon ), 1)))
FROM   counties AS c1, counties AS c2;
```

La conversión de tipos sólo es necesaria cuando se pasa de un supertipo de la jerarquía a un subtipo. Para obtener más información sobre el tratamiento de subtipos, vea el manual *Consulta de SQL*.

AsShape

AsShape utiliza como entrada un objeto de geometría y devuelve un BLOB.

Sintaxis

db2gse.AsShape(g db2gse.ST_Geometry)

Tipo devuelto

BLOB(1m)

Ejemplos

El siguiente fragmento de código muestra cómo la función AsShape convierte los polígonos de zona de la tabla SENSITIVE_AREAS en polígonos de forma. Estos polígonos de forma se pasan a la función draw_polygon de la aplicación para su visualización.

```
/* Crear la expresión SQL. */
strcpy(sqlstmt, "select db2gse.AsShape (zone) from SENSITIVE_AREAS
where db2gse.EnvelopesIntersect(zone, db2gse.PolyFromShape(cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Preparar la sentencia SQL. */
SQLPrepare(hstmt, (UCHAR *)sqlstmt, SQL_NTS);

/* Definir la longitud pcbvalue1 de la forma. */
pcbvalue1 = blob_len;

/* Vincular el parámetro de forma. */
SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_BINARY, SQL_BLOB, blob_len,
0, shape, blob_len, &pcbvalue1);

/* Ejecutar la consulta. */
rc = SQLExecute(hstmt);

/* Asignar los resultados de la consulta, (los polígonos de zona) a la
   variable fetched_binary. */
SQLBindCol (hstmt, 1, SQL_C_Binary, fetched_binary, 100000, &ind_blob);

/* Buscar y cargar cada polígono dentro de la ventana de visualización y
mostrarlo. */

while(SQL_SUCCESS == (rc = SQLFetch(hstmt)))
  draw_polygon(fetched_binary);
```

EnvelopesIntersect

EnvelopesIntersect devuelve un 1 (TRUE) si las envolventes de las dos geometrías forman intersección y 0 (FALSE) si no es así.

Sintaxis

db2gse.EnvelopesIntersect(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La función get_window recupera las coordenadas de la ventana de visualización a partir de la aplicación. El parámetro de ventana es realmente una estructura de forma de polígono que contiene una serie de coordenadas que representan el polígono de visualización. La función PolyFromShape convierte la forma de ventana de visualización en un polígono de Spatial Extender que la función EnvelopesIntersect utiliza como su envolvente de intersección. Se devuelven todos los polígonos de la zona SENSITIVE_AREAS que forman intersección con el interior o el límite de la ventana de visualización. Cada polígono se obtiene del conjunto resultante y se pasa a la función draw_polygon.

```
/* Obtener las coordenadas de la ventana de visualización como una forma de
polígono. */
get_window(&window)

/* Crear la expresión SQL. Se utilizará la función db2gse.EnvelopesIntersect
  para limitar el conjunto resultante a los polígonos de zona que formen
  intersección con la envolvente de la ventana de visualización. */
strcpy(sqlstmt, "select db2gse.AsShape(zone) from SENSITIVE_AREAS where
db2gse.EnvelopesIntersect (zone, db2gse.PolyFromShape(cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Definir blob_len en la longitud de bytes de un polígono de forma de 5
puntos. */
blob_len = 128;

/* Preparar la sentencia SQL. */
SQLPrepare(hstmt, (UCHAR *)sqlstmt, SQL_NTS);

/* Definir pcbvalue1 en la forma de la ventana */
pcbvalue1 = blob_len;

/* Vincular el parámetro de forma. */
SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_BINARY, SQL_BLOB,
blob_len,0, window, blob_len, &pcbvalue1);

/* Ejecutar la consulta. */
rc = SQLExecute(hstmt);

/* Asignar los resultados de la consulta, (los polígonos de zona) a la
   variable fetched_binary. */
```

```
SQLBindCol (hstmt, 1, SQL_C_Binary, fetched_binary, 100000, &ind_blob);

/* Buscar y cargar cada polígono dentro de la ventana de visualización y
mostrarlo. */
while(SQL_SUCCESS == (rc = SQLFetch(hstmt))
  draw_polygon(fetched_binary);
```

GeometryFromShape

GeometryFromShape utiliza como entrada una forma y un identificador de sistema de referencias espaciales, y devuelve un objeto de geometría.

Sintaxis

db2gse.GeometryFromShape(ShapeGeometry Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

El siguiente fragmento de código C contiene funciones ODBC intercaladas con funciones SQL de Spatial Extender que insertan datos en la tabla LOTS.

La tabla LOTS se creó con dos columnas: la columna LOT_ID, que identifica de forma exclusiva cada parcela, y la columna de polígonos LOT, que contiene la geometría de cada parcela.

```
CREATE TABLE LOTS ( lot_id  integer,
                    lot     db2gse.ST_Polygon);
```

La función GeometryFromShape convierte formas a una geometría de Spatial Extender. La sentencia INSERT completa se copia en shp_sql. La sentencia INSERT contiene marcadores de parámetro para aceptar los datos LOT_ID y los datos LOT de forma dinámica.

```
/* Crear la sentencia INSERT de SQL para entrar datos en las columnas
lot_id y lot. Los interrogantes son marcadores de parámetros que indican
los valores de lot_id y de lot que se recuperarán durante la ejecución. */
strcpy (shp_sql,"insert into LOTS (lot_id, lot) values (?,
db2gse.GeometryFromShape (cast(? as blob(1m)), db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor de clave de entero con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_SLONG,
     SQL_INTEGER, 0, 0, &lot_id, 0, &pcbvalue1);

/* Vincular la forma con el segundo parámetro. */
pcbvalue2 = blob_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, blob_len, 0, shape_blob, blob_len, &pcbvalue2);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

Is3d

Is3d utiliza como entrada un objeto de geometría y devuelve un 1 (TRUE) si el objeto tiene coordenadas 3D y 0 (FALSE) si no es así.

Sintaxis

db2gse.Is3d(g db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla THREED_TEST, que tiene dos columnas, la columna GID de tipo entero y la columna de geometría G1.

```
CREATE TABLE THREED_TEST (gid smallint, g1 db2gse.ST_Geometry)
```

Las sentencias INSERT insertan dos puntos en la tabla THREED_TEST. El primer punto no contiene coordenadas Z y el segundo sí.

```
INSERT INTO THREED_TEST
VALUES(1, db2gse.ST_PointFromText('point (10 10)', db2gse.coordref()..srid(0)))

INSERT INTO THREED_TEST
VALUES (2, db2gse.ST_PointFromText('point z (10.92 10.12 5)',
db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT lista el contenido de la columna GID con los resultados de la función Is3d. La función devuelve un 0 para la primera fila, que no contiene coordenadas Z y un 1 para la segunda fila, que sí contiene coordenadas Z.

```
SELECT gid, db2gse.Is3d (g1) "Is it 3d?" FROM THREED_TEST
```

Se devuelve el siguiente conjunto resultante.

```
gid      Is it 3d?
------   ----------
     1            0
     2            1
```

IsMeasured

IsMeasured utiliza como entrada un objeto de geometría y devuelve un 1 (TRUE) si el objeto tiene medidas y 0 (FALSE) si no es así.

Sintaxis

db2gse.IsMeasured(g db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla MEASURE_TEST, que tiene dos columnas. La columna GID identifica de forma exclusiva las filas y la columna G1 guarda las geometrías de punto.

```
CREATE TABLE MEASURE_TEST (gid smallint, g1 db2gse.ST_Geometry)
```

Las siguientes sentencias INSERT insertan dos registros en la tabla MEASURE_TEST. El primer registro guarda un punto que no tiene medida. El punto del segundo registro sí tiene una medida.

```
INSERT INTO MEASURE_TEST
VALUES(1, db2gse.ST_PointFromText('point (10 10)', db2gse.coordref()..srid(0)))

INSERT INTO MEASURE_TEST
VALUES (2, db2gse.ST_PointFromText('point m (10.92 10.12 5)',
    db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente muestran la columna GID junto con los resultados de la función IsMeasured. La función IsMeasured devuelve un 0 para la primera fila porque el punto no tiene medida. Devuelve un 1 para la segunda fila porque el punto tiene medidas.

```
SELECT gid, db2gse.IsMeasured (g1) "Has measures?" FROM MEASURE_TEST
gid      Has measures
------   ----------
     1            0
     2            1
```

LineFromShape

LineFromShape utiliza como entrada una forma de tipo punto y un identificador de sistema de referencias espaciales, y devuelve una línea.

Sintaxis

db2gse.Line FromShape(ShapeLineString Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_LineString

Ejemplos

El siguiente fragmento de código llena la tabla SEWERLINES con el id exclusivo, clase de tamaño y geometría de cada línea de alcantarillado.

La sentencia CREATE TABLE crea la tabla SEWERLINES, que tiene tres columnas. La primera columna, SEWER_ID, identifica de forma exclusiva cada línea de alcantarillado. La segunda columna, CLASS, de tipo entero, identifica el tipo de línea de alcantarillado, que suele estar asociado a la capacidad de la línea. La tercera columna, SEWER, de tipo línea, guarda la geometría de la línea de alcantarillado.

```
CREATE TABLE SEWERLINES (sewer_id   integer, class  integer,
                         sewer db2gse.ST_LineString);

/* Crear la sentencia SQL insert para llenar el id de alcantarillado,
   la clase de tamaño y la línea del alcantarillado.
   Los interrogantes son marcadores de parámetro que indican los valores
   de id de alcantarillado, clase y geometría de alcantarillado que se
   recuperarán en el momento de la ejecución. */
strcpy (shp_sql,"insert into sewerlines (sewer_id,class,sewer)
values (?,?, db2gse.Line FromShape (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor de clave de entero con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_SLONG,
   SQL_INTEGER, 0, 0, &sewer_id, 0, &pcbvalue1);

/* Vincular el valor entero class con el segundo parámetro. */
pcbvalue2 = 0;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_SLONG,
   SQL_INTEGER, 0, 0, &sewer_class, 0, &pcbvalue2);;

/* Vincular la forma con el tercer parámetro. */
pcbvalue3 = blob_len;
```

```
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, blob_len, 0, sewer_shape, blob_len, &pcbvalue3);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

LocateAlong

LocateAlong utiliza como entrada una geometría y una medida, y devuelve como multipunto el grupo de puntos encontrados en la medida.

Si LocateAlong utiliza como entrada un multipunto y una medida, y la medida no está incluida en el multipunto, entonces LocateAlong devuelve POINT EMPTY.

Sintaxis

db2gse.LocateAlong(g db2gse.ST_Geometry, [medida] Double)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla LOCATEALONG_TEST. LOCATEALONG_TEST tiene dos columnas: la columna GID, que identifica de forma exclusiva cada fila, y la columna de geometría G1, que guarda la geometría de ejemplo.

```
CREATE TABLE LOCATEALONG_TEST (gid integer, g1 db2gse.ST_Geometry)
```

Las siguientes sentencias INSERT insertan dos filas. La primera es una multilínea; la segunda es un multipunto.

```
INSERT INTO db2gse.LOCATEALONG_TEST VALUES(
1, db2gse.ST_MLineFromText('multilinestring m ((10.29 19.23 5,23.82 20.29 6,
                                        30.19 18.47 7,
45.98 20.74 8), (23.82 20.29 6,30.98 23.98 7,42.92 25.98 8))',
                                  db2gse.coordref()..srid(0)))

INSERT INTO db2gse.LocateAlong_TEST VALUES(
2, db2gse.ST_MPointFromText('multipoint m (10.29 19.23 5,23.82 20.29 6,
 30.19 18.47 7,45.98 20.74 8,23.82 20.29 6,30.98 23.98 7,42.92 25.98)',
                 db2gse.coordref()..srid(0)))
```

En la siguiente sentencia SELECT y el correspondiente conjunto resultante, se indica a la función LocateAlong que busque puntos cuya medida es 6,5. La primera fila devuelve un multipunto que contiene dos puntos. Sin embargo, la segunda fila devuelve un punto vacío. Para elementos geográficos lineales (geometría con una dimensión mayor que 0), LocateAlong puede interpolar el punto; sin embargo, para multipuntos, la medida resultante debe coincidir exactamente.

```
SELECT gid, CAST(db2gse.ST_AsText(db2gse.LocateAlong (g1,6.5)) AS
varchar(96))
"Geometry"
FROM LOCATEALONG_TEST
```

```
GID         Geometry
----------- ------------------------------------------------------------

          1 MULTIPOINT M ( 27.01000000 19.38000000 6.50000000, 27.40000000
22.14000000 6.50000000)
          2 POINT EMPTY

  2 record(s) selected.
```

En la siguiente sentencia SELECT y el correspondiente conjunto resultante, la función LocateAlong devuelve líneas multipunto para ambas filas. La medida resultante 7 coincide con las medidas de los datos de entrada de tipo multilínea y multipunto.

```
SELECT gid,CAST(db2gse.ST_AsText(db2gse.LocateAlong (g1,7)) AS varchar(96))
   "Geometry"
FROM LOCATEALONG_TEST

GID         Geometry
----------- ------------------------------------------------------------
          1 MULTIPOINT M ( 30.19000000 18.47000000 7.00000000, 30.98000000
 23.98000000 7.00000000)
          2 MULTIPOINT M ( 30.19000000 18.47000000 7.00000000, 30.98000000
 23.98000000 7.00000000)

  2 record(s) selected.
```

LocateBetween

LocateBetween utiliza como entrada un objeto de geometría y dos ubicaciones de medidas y devuelve una geometría que representa el grupo de rutas desconectadas entre las dos ubicaciones de medidas.

Sintaxis

db2gse.LocateBetween(g db2gse.ST_Geometry, [medida] Double, [medida]5 Double)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla LOCATEBETWEEN_TEST. LOCATEBETWEEN_TEST tiene dos columnas: la columna GID, que identifica de forma exclusiva cada fila, y la columna de multilínea G1, que guarda la geometría de ejemplo.

```
CREATE TABLE LOCATEBETWEEN_TEST (gid integer, g1 db2gse.ST_Geometry)
```

Las siguientes sentencias INSERT insertan dos filas en la tabla LOCATEBETWEEN_TEST. La primera fila es una multilínea y la segunda es un multipunto.

```
INSERT INTO db2gse.LOCATEBETWEEN_TEST
VALUES(1,db2gse.ST_MLineFromText('multilinestring m ((10.29 19.23 5,
                           23.82 20.29 6, 30.19 18.47 7,45.98 20.74 8),
                                        (23.82 20.29 6,30.98 23.98 7,
                                         42.92 25.98 8))',
                      db2gse.coordref()..srid(0)))

INSERT INTO db2gse.LOCATEBETWEEN_TEST
VALUES(2, db2gse.ST_MPointFromText('multipoint m (10.29 19.23 5,23.82 20.29 6,
30.19 18.47 7,45.98 20.74 8,23.82 20.29 6,
30.98 23.98 7,42.92 25.98 8)',
                        db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente muestran cómo la función LocateBetween localiza las medidas que quedan entre las medidas 6,5 y 7,5, ambas inclusive. La primera fila devuelve una multilínea que contiene varias líneas. La segunda fila devuelve un multipunto, pues los datos de entrada eran un multipunto. Cuando los datos de entrada tienen una dimensión de 0 (punto o multipunto), es necesaria una coincidencia exacta.

```
SELECT gid, CAST(db2gse.ST_AsText(db2gse.LocateBetween (g1,6.5,7.5))
       AS varchar(96)) "Geometry"
FROM LOCATEBETWEEN_TEST
```

```
GID         Geometry
----------- ------------------------------------------------------------------
          1 MULTILINESTRING M ( 27.01000000 19.38000000 6.50000000, 31.19000000
18.47000000 7.00000000,38.09000000 19.61000000 7.50000000),(27.40000000 22.1400
0000 6.50000000, 30.98000000 23.98000000 7.00000000,36.95000000 24.98000000 7.5
0000000)
          2 MULTIPOINT M ( 30.19000000 18.47000000 7.00000000, 30.98000000 23.9
8000000 7.00000000)

  2 record(s) selected.
```

M

M utiliza como entrada un punto y devuelve su medida.

Sintaxis

db2gse.M(p db2gse.ST_Point)

Tipo devuelto

Double

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla M_TEST. M_TEST tiene dos columnas: la columna de entero GID, que identifica de forma exclusiva la fila, y la columna de punto PT1, que guarda la geometría de ejemplo.

```
CREATE TABLE M_TEST (gid integer, pt1 db2gse.ST_Point)
```

Las siguientes sentencias INSERT insertan una fila que contiene un punto con medidas y una fila que contiene un punto sin medidas.

```
INSERT INTO db2gse.M_TEST
VALUES(1, db2gse.ST_PointFromText('point (10.02 20.01)',
db2gse.coordref()..srid(0)))
INSERT INTO db2gse.M_TEST
VALUES(2, db2gse.ST_PointFromText('point  zm(10.02 20.01 5.0 7.0)',
db2gse.coordref()..srid(0)))
```

En la siguiente sentencia SELECT y el conjunto resultante correspondiente, la función M lista los valores de medida de los puntos. Puesto que el primer punto no tiene medidas, la función M devuelve un NULO.

```
SELECT gid, db2gse.M (pt1) "The measure" FROM M_TEST

GID         The measure
----------- ------------------------
          1                        -
          2   +7.00000000000000E+000

  2 record(s) selected.
```

MLine FromShape

MLine FromShape utiliza como entrada una forma de tipo multilínea y un identificador de sistema de referencias espaciales, y devuelve una multilínea.

Sintaxis

db2gse.MLineFromShape(ShapeMultiLineString Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiLineString

Ejemplos

El siguiente fragmento de código llena la tabla WATERWAYS con un id exclusivo, un nombre y una multilínea denominada WATER.

La tabla WATERWAYS se crea con las columnas ID y NAME que identifican cada sistema de corrientes y de ríos guardado en la tabla. La columna WATER es una multilínea, pues los sistemas de ríos y de corrientes suelen ser un agregado de varias líneas.

```
CREATE TABLE WATERWAYS (id        integer,
                        name       varchar(128),
                        water      db2gse.ST_MultiLineString);

/* Crear la sentencia INSERT de SQL para entrar datos en el
ID, el nombre y la multilínea. Los interrogantes son marcadores de
parámetros que indican el ID, el nombre y los valores de WATER que se
recuperarán durante la ejecución. */
strcpy (shp_sql,"insert into WATERWAYS (id,name,water)
values (?,?, db2gse.MLineFromShape (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor entero id con el primer parámetro. */

pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_SLONG,
   SQL_INTEGER, 0, 0, &id, 0, &pcbvalue1);
/* Vincular el valor varchar name con el segundo parámetro. */

pcbvalue2 = name_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_CHAR,
     SQL_CHAR, name_len, 0, &name, name_len, &pcbvalue2);

/* Vincular la forma con el tercer parámetro. */
pcbvalue3 = blob_len;
```

```
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, blob_len, 0, water_shape, blob_len, &pcbvalue3);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

MPointFromShape

MPointFromShape utiliza como entrada una forma de tipo multipunto y un identificador de sistema de referencias espaciales, y devuelve un multipunto.

Sintaxis

db2gse.MPointFromShape(ShapeMultiPoint (1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiPoint

Ejemplos

Este fragmento de código llena una tabla SPECIES_SITINGS de biología.

La tabla SPECIES_SITINGS se crea con tres columnas. Las columnas SPECIES y GENUS identifican de forma exclusiva cada fila, mientras que el multipunto SITINGS guarda las ubicaciones de las especies.

```
CREATE TABLE SPECIES_SITINGS (species  varchar(32),
                              genus varchar(32),
                              sitings  db2gse.ST_MultiPoint);

/* Crear la sentencia SQL insert para llenar las columnas species, genus y
   sitings. Los interrogantes son marcadores de parámetro que indican los
   valores de name y water que se recuperarán en el momento de la ejecución.*/
strcpy (shp_sql,"insert into SPECIES_SITINGS (species,genus,sitings)
values (?,?, db2gse.MPointFromShape (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor varchar species con el primer parámetro. */
pcbvalue1 = species_len;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_CHAR,
     SQL_CHAR, species_len, 0, species, species_len, &pcbvalue1);

/* Vincular el valor varchar genus con el segundo parámetro. */
pcbvalue2 = genus_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_CHAR,
   SQL_CHAR, genus_len, 0, name, genus_len, &pcbvalue2);

/* Vincular la forma con el tercer parámetro. */
pcbvalue3 = blob_len;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, sitings_len, 0, sitings_shape, sitings_len, &pcbvalue3);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

MPolyFromShape

MPolyFromShape utiliza como entrada una forma de tipo multipolígono y un identificador de sistema de referencias espaciales, y devuelve un multipolígono.

Sintaxis

db2gse.MPolyFromShape(ShapeMultiPolygon Blob(1m), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiPolygon

Ejemplos

Este fragmento de código llena la tabla LOTS.

La tabla LOTS guarda los ID de parcela que identifican de forma exclusiva cada parcela y el multipolígono de parcela que contiene la geometría lineal de la parcela.

```
CREATE TABLE LOTS (lot_id   integer, lot   db2gse.ST_MultiPolygon);

/* Crear la sentencia SQL insert para llenar las columnas lot_id y lot. Los
   interrogantes son marcadores de parámetro que indican los valores lot_id
   y lot que se recuperarán en el momento de la ejecución.*/
strcpy (shp_sql,"insert into LOTS (lot_id,lot)
values (?, db2gse.MPolyFromShape (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor entero lot_id con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_INTEGER,
     SQL_INTEGER, 0, 0, &lot_id, 0, &pcbvalue1);

/* Vincular la forma de parcela con el segundo parámetro. */
pcbvalue2 = lot_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, lot_len, 0, lot_shape, lot_len, &pcbvalue2);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

PointFromShape

PointFromShape utiliza como entrada una forma de tipo punto y un identificador de sistema de referencias espaciales, y devuelve un punto.

Sintaxis

db2gse.PointFromShape(db2gse.ShapePoint blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Point

Ejemplos

El fragmento del programa llena la tabla HAZARDOUS_SITES.

Los sitios de riesgo se guardan en la tabla HAZARDOUS_SITES, creada con la siguiente sentencia CREATE TABLE. La columna location, definida como un punto, guarda una ubicación que representa el centro geográfico de cada sitio peligroso.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(128),
                              location  db2gse.ST_Point);

/* Crear la sentencia SQL insert para llenar las columnas site_id, name y
   location. Los interrogantes son marcadores de parámetro que indican los
   valores de site_id, name y location que se recuperarán en el momento de
   la ejecución. */
strcpy (shp_sql,"insert into HAZARDOUS_SITES (site_id, name, location)
values (?,?, db2gse.PointFromShape (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor entero site_id con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_INTEGER,
     SQL_INTEGER, 0, 0, &site_id, 0, &pcbvalue1);

/* Vincular el valor varchar name con el segundo parámetro. */
pcbvalue2 = name_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_CHAR,
   SQL_CHAR, 0, 0, name, 0, &pcbvalue2);

/* Vincular la forma location con el tercer parámetro. */
pcbvalue3 = location_len;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
```

```
        SQL_BLOB, location_len, 0, location_shape, location_len, &pcbvalue3);

    /* Ejecutar la sentencia insert. */
    rc = SQLExecute (hstmt);
```

PolyFromShape

PolyFromShape utiliza como entrada una forma de tipo polígono y un identificador de sistema de referencias espaciales, y devuelve un polígono.

Sintaxis

db2gse.PolyFromShape (ShapePolygon Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Polygon

Ejemplos

El fragmento del programa llena la tabla SENSITIVE_AREAS. Los interrogantes representan marcadores de parámetro para los valores id, name, size, type y zone que se recuperarán en el momento de la ejecución.

La tabla SENSITIVE_AREAS contiene varias columnas que describen las instituciones amenazadas además de la columna ZONE, que guarda la geometría polígono de la institución.

```
CREATE TABLE SENSITIVE_AREAS (id        integer,
                              name      varchar(128),
                              size      float,
                              type      varchar(10),
                              zone      db2gse.ST_Polygon);

/* Crear la sentencia SQL insert para llenar las columnas
   id, name, size, type y zone. Los interrogantes son marcadores
   de parámetro que indican los valores id, name, size, type y zone
   que se recuperarán en el momento de la ejecución. */
strcpy (shp_sql,"insert into SENSITIVE_AREAS (id, name, size, type, zone)
values (?,?,?,?, db2gse.PolyFromShape (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor entero id con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_INTEGER,
     SQL_INTEGER, 0, 0, &site_id, 0, &pcbvalue1);
/* Vincular el valor varchar name con el segundo parámetro. */
pcbvalue2 = name_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_CHAR,
   SQL_CHAR, 0, 0, name, 0, &pcbvalue2);

/* Vincular el valor float size con el tercer parámetro. */
pcbvalue3 = 0;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_FLOAT,
```

```
     SQL_REAL, 0, 0, &size, 0, &pcbvalue3);

/* Vincular el valor varchar type con el cuarto parámetro. */
pcbvalue4 = type_len;
rc = SQLBindParameter (hstmt, 4, SQL_PARAM_INPUT, SQL_C_CHAR,
     SQL_VARCHAR, type_len, 0, type, type_len, &pcbvalue4);

/* Vincular el polígono zone con el quinto parámetro. */
pcbvalue5 = zone_len;
rc = SQLBindParameter (hstmt, 5, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, zone_len, 0, zone_shp, zone_len, &pcbvalue5);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

ShapeToSQL

ShapeToSQL construye un valor db2gse.ST_Geometry a partir de su representación de forma. El valor SRID de 0 se utiliza automáticamente.

Sintaxis

db2gse.ShapeToSQL(ShapeGeometry blob(1M))

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

El siguiente fragmento de código C contiene funciones ODBC intercaladas con funciones SQL de Spatial Extender que insertan datos en la tabla LOTS. La tabla LOTS se creó con dos columnas: la columna lot_id, que identifica de forma exclusiva cada parcela, y la columna lot, de tipo multipolígono, que contiene la geometría de cada parcela.

```
CREATE TABLE lots (lot_id  integer,
                   lot     db2gse.ST_MultiPolygon);
```

La función ShapeToSQL convierte formas en una geometría de Spatial Extender. La sentencia INSERT completa se copia en shp_sql. La sentencia INSERT contiene marcadores de parámetro para aceptar los datos LOT_id y los datos lot de forma dinámica.

```
/* Crear la sentencia insert de SQL para entrar datos en el ID de
parcela y el multipolígono de parcela. Los interrogantes son marcadores
de parámetro que indican el ID de parcela y los valores de parcela que
se recuperarán durante la ejecución. */

strcpy (shp_sql,"insert into lots (lot_id, lot) values(?,
db2gse.ShapeToSQL(cast(? as blob(1m))))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/

rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */

rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor de clave de entero con el primer parámetro. */

pcbvalue1 = 0;

rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_SLONG,
     SQL_INTEGER, 0, 0, &lot_id, 0, &pcbvalue1);

/* Vincular la forma con el segundo parámetro. */

pcbvalue2 = blob_len;
```

```
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, blob_len, 0, shape_blob, blob_len, &pcbvalue2);

/* Ejecutar la sentencia insert. */

rc = SQLExecute (hstmt);
```

ST_Area

ST_Area utiliza como entrada un polígono o multipolígono, y devuelve su superficie.

Sintaxis

db2gse.ST_Area(s db2gse.ST_Surface)
db2gse.ST_Surface
db2gse.ST_Polygondb2gse.ST_MultiSurface
db2gse.ST_MultiPolygon

Tipo devuelto

Double

Ejemplos

Un ingeniero municipal necesita una lista de las zonas edificadas. Para obtener la lista, un técnico de GIS selecciona el ID de edificio y el área que ocupa cada edificio.

Las zonas edificadas se guardas en la tabla BUILDINGFOOTPRINTS que se creó con la siguiente sentencia CREATE TABLE:

```
CREATE TABLE BUILDINGFOOTPRINTS (   building_id integer,
                                    lot_id      integer,
                                    footprint   db2gse.ST_MultiPolygon);
```

Para satisfacer la petición del ingeniero municipal, el técnico utiliza la siguiente sentencia SELECT para seleccionar la clave exclusiva, el id de edificio y el área de cada zona edificada de la tabla BUILDINGFOOTPRINTS:

```
SELECT building_id, db2gse.ST_Area (footprint) "Area"
  FROM BUILDINGFOOTPRINTS;
```

La sentencia SELECT devuelve el siguiente conjunto resultante:

```
building_id   Area
-----------  ------------------------
        506    +1.40768000000000E+003
       1208    +2.55759000000000E+003
        543    +1.80786000000000E+003
        178    +2.08671000000000E+003
        .
        .
        .
```

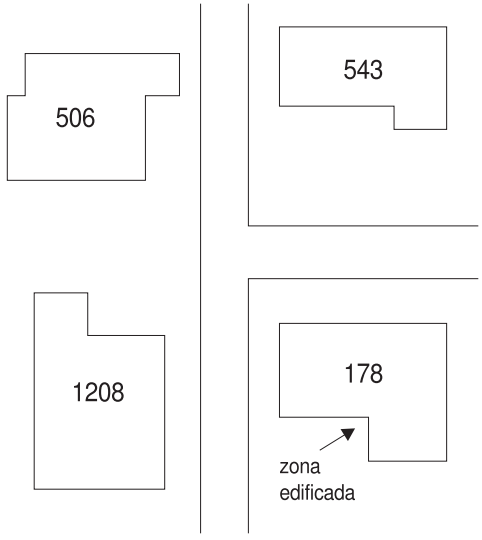


Figura 28. Utilización del área para encontrar una zona edificada. Cuatro de las zonas edificadas y los números de ID de sus correspondientes edificios se muestran junto con la calle adyacente.

Nota: Tal como se indicó anteriormente, ST_Area puede utilizar como entrada un polígono o un multipolígono. Por tanto, el tipo de datos del parámetro de entrada puede ser:

- Cualquiera de los dos tipos de datos utilizados para los polígonos: (db2gse.ST_Surface o db2gse.ST_Polygon)
- Cualquiera de los dos tipos de datos utilizados para los multipolígonos: (db2gse.ST_MultiSurface o db2gse.ST_MultiPolygon)

ST_AsBinary

ST_AsBinary utiliza como entrada un objeto de geometría y devuelve su representación binaria convencional. ST_AsBinary no puede utilizar como entrada una geometría vacía (SQLSTATE 38827).

Sintaxis

db2gse.ST_AsBinary(g db2gse.ST_Geometry)

Tipo devuelto

BLOB(1m)

Ejemplos

El siguiente fragmento de código muestra cómo la función ST_AsBinary convierte los multipolígonos de zona edificada de la tabla BUILDINGFOOTPRINTS en multipolígonos WKB. Estos multipolígonos se pasan a la función draw_polygon de la aplicación para su visualización.

```
/* Crear la expresión SQL. */
strcpy(sqlstmt, "select db2gse.ST_AsBinary (footprint) from BUILDINGFOOTPRINTS
where db2gse.EnvelopesIntersect(footprint, db2gse.ST_PolyFromWKB
(cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Preparar la sentencia SQL. */
SQLPrepare(hstmt, (UCHAR *)sqlstmt, SQL_NTS);

/* Definir la longitud pcbvalue1 de la forma. */
pcbvalue1 = blob_len;

/* Vincular el parámetro de forma. */
SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_BINARY, SQL_BLOB, blob_len,
0, shape, blob_len, &pcbvalue1);

/* Ejecutar la consulta. */
rc = SQLExecute(hstmt);

/* Asignar los resultados de la consulta, (los polígonos de zona) a la
  variable fetched_binary.
 */
SQLBindCol (hstmt, 1, SQL_C_Binary, fetched_binary, 100000, &ind_blob);

/* Buscar y cargar cada polígono dentro de la ventana de visualización y
mostrarlo. */
while(SQL_SUCCESS == (rc = SQLFetch(hstmt)))
  draw_polygon(fetched_binary);
```

ST_AsText

db2gse.ST_AsText utiliza como entrada un objeto de geometría y devuelve su representación de texto convencional.

Sintaxis

db2gse.ST_AsText(g db2gse.ST_Geometry)

Tipo devuelto

Varchar(4000)

Ejemplos

En el siguiente caso, la función db2gse.ST_AsText convierte el punto de ubicación HAZARDOUS_SITES en su representación de texto.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(40),
                              location  db2gse.ST_Point);
INSERT INTO HAZARDOUS_SITES
VALUES (102,
        'W. H. Kleenare Chemical Repository',
        db2gse.ST_PointFromText('point (1020.12 324.02)',
db2gse.coordref()..srid(0)));

SELECT site_id, name, cast(db2gse.ST_AsText(location) as varchar(40))
    "Location"
FROM HAZARDOUS_SITES;
```

La sentencia SELECT devuelve el siguiente conjunto resultante:

```
SITE_ID Name                                Location
------- ----------------------------------- ---------------------------------
    102 W. H. Kleenare Chemical Repository  POINT (1020.00000000 324.00000000)
```

ST_Boundary

ST_Boundary utiliza como entrada un objeto de geometría y devuelve su límite combinado como un objeto de geometría. Los puntos y multipuntos dan siempre como resultado un límite que es una geometría vacía de dimensión 0 (no de dimensión 1).

Sintaxis

db2gse.ST_Boundary(g db2gse.ST_Geometry)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

En el siguiente fragmento de código, se crea una tabla denominada BOUNDARY_TEST. BOUNDARY_TEST tiene dos columnas: GEOTYPE, que se define como varchar, y G1, que se define como la geometría de superclase. Las siguientes sentencias INSERT insertan cada una de las geometrías de subclase. La función ST_Boundary recupera el límite de cada subclase guardada en la columna de geometría G1. Observe que la dimensión de la geometría resultante es siempre un nivel menor que la geometría de entrada. Los puntos y los multipuntos siempre dan como resultado un límite que una geometría vacía, dimensión 1. Las líneas y las multilíneas devuelven un límite de multipunto, dimensión 0. Un polígono o un multipolígono devuelve siempre un límite de multilínea, dimensión 1.

```
CREATE TABLE BOUNDARY_TEST (GEOTYPE varchar(20), G1 db2gse.ST_Geometry)

INSERT INTO BOUNDARY_TEST
VALUES('Point',
       db2gse.ST_PointFromText('point (10.02 20.01)',
                     db2gse.coordref()..srid(0)))

INSERT INTO BOUNDARY_TEST
VALUES('Linestring',
       db2gse.ST_LineFromText('linestring  (10.02 20.01,10.32 23.98,
11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO BOUNDARY_TEST
VALUES('Polygon',
       db2gse.ST_PolyFromText('polygon ((10.02 20.01,11.92 35.64,25.02 34.15,
                                  19.15 33.94, 10.02 20.01))',
                     db2gse.coordref()..srid(0)))

INSERT INTO BOUNDARY_TEST
VALUES('Multipoint',
       db2gse.ST_MPointFromText('multipoint (10.02 20.01,10.32 23.98,
11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO BOUNDARY_TEST
VALUES('Multilinestring',
       db2gse.ST_MLineFromText('multilinestring ((10.02 20.01,10.32 23.98,
```

```
11.92 25.64), (9.55 23.75,15.36 30.11))',
                    db2gse.coordref()..srid(0)))

INSERT INTO BOUNDARY_TEST
VALUES('Multipolygon',
       db2gse.ST_MPolyFromText('multipolygon (((10.02 20.01,11.92 35.64,
        25.02 34.15, 19.15 33.94,10.02 20.01)),
                                      ((51.71 21.73,73.36 27.04,71.52 32.87,
                                        52.43 31.90,51.71 21.73)))',
                    db2gse.coordref()..srid(0)))

SELECT GEOTYPE,
       CAST(db2gse.ST_AsText(db2gse.ST_Boundary (G1)) as varchar(280))
      "The boundary"
FROM BOUNDARY_TEST


GEOTYPE              The boundary
-------------------- ------------------------------------------------------------
Point                POINT EMPTY
Linestring           MULTIPOINT ( 10.02000000 20.01000000, 11.92000000
 25.64000000)
Polygon              MULTILINESTRING (( 10.02000000 20.01000000, 19.15000000
 33.94000000, 25.02000000 34.15000000, 11.92000000 35.64000000, 10.02000000
 20.01000000))
Multipoint           POINT EMPTY
Multilinestring      MULTIPOINT ( 9.55000000 23.75000000, 10.02000000
 20.01000000, 11.92000000 25.64000000, 15.36000000 30.11000000)
Multipolygon         MULTILINESTRING (( 51.71000000 21.73000000, 73.36000000
 27.04000000, 71.52000000 32.87000000, 52.43000000 31.90000000, 51.71000000
 21.73000000),( 10.02000000 20.01000000, 19.15000000 33.94000000, 25.02000000
 34.15000000, 11.92000000 35.64000000, 10.02000000 20.01000000))

  6 record(s) selected.
```

ST_Buffer

ST_Buffer utiliza como entrada una geometría y una distancia y devuelve la geometría que rodea al objeto original.

Sintaxis

db2gse.ST_Buffer(g db2gse.ST_Geometry , [medida] Double)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

Supongamos que el supervisor de un condado necesita una lista de lugares peligrosos cuyo radio de cinco millas solape áreas especialmente sensibles, como colegios, hospitales y guarderías. Las áreas sensibles se guardan en la tabla SENSITIVE_AREAS que se crea con la siguiente sentencia CREATE TABLE. La columna ZONE está definida como un polígono, que se guarda como el contorno de cada área sensible.

```
CREATE TABLE SENSITIVE_AREAS (id    integer,
                              name  varchar(128),
                              size  float,
                              type  varchar(10),
                              zone  db2gse.ST_Polygon);
```

Los sitios peligrosos se guardan en la tabla HAZARDOUS_SITES que se crea con la siguiente sentencia CREATE TABLE. La columna LOCATION, definida como un punto, almacena la ubicación que constituye el centro geográfico de cada sitio peligroso.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(128),
                              location  db2gse.ST_Point);
INSERT INTO HAZARDOUS SITES
VALUES (102, 'Allied Chemicals',
        db2gse.ST_PointFromText('Point(157000 475000)',coordref()..srid(0)))
```

Las tablas SENSITIVE_AREAS y HAZARDOUS_SITES se unen mediante la función db2gse.ST_Overlaps. La función devuelve un 1 (TRUE) para todas las filas de SENSITIVE_AREAS cuyos polígonos de zona se solapan con el radio de cinco millas que rodea el punto de la ubicación HAZARDOUS_SITES.

```
SELECT sa.name "Sensitive Areas", hs.name "Hazardous Sites"
FROM SENSITIVE_AREAS sa, HAZARDOUS_SITES hs
WHERE db2gse.ST_Overlaps(sa.zone, db2gse.ST_Buffer (hs.location,(5 * 5280)))
=  1;
```

(5 * 5280 denota cinco millas. Esto deriva del hecho de que una milla tiene 5280 pies, y que un pie es la unidad lineal del sistema de coordenadas al que pertenecen las coordenadas especificadas en la sentencia VALUES).

En la Figura 29, algunas de las áreas sensibles de este distrito administrativo quedan dentro de las cinco millas que rodean las ubicaciones de los sitios peligrosos. Ambas zonas que rodean con un radio de cinco millas forman intersección con el hospital, y una forma intersección con el colegio. Sin embargo, la guardería queda fuera de ambos radios.

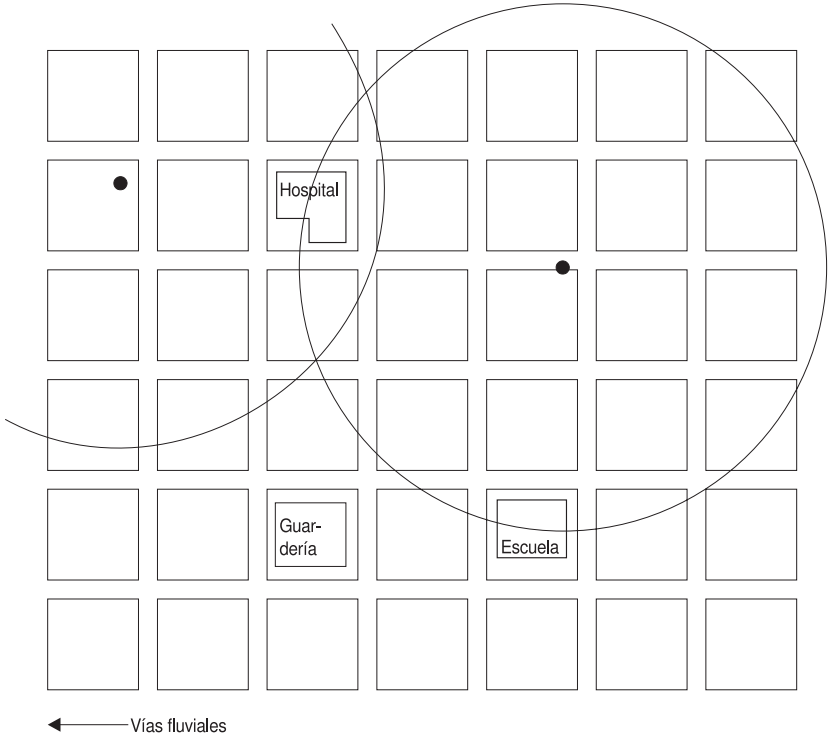


Figura 29. Se redondea un punto con un radio de cinco millas

ST_Centroid

ST_Centroid utiliza como entrada un polígono o multipolígono y devuelve su centro geométrico en forma de punto.

Sintaxis

db2gse.ST_Centroid(s db2gse.ST_Surface)
db2gse.ST_Centroid(ms db2gse.ST_MultiSurface)

Tipo devuelto

Para superficie: db2gse.ST_Point

Ejemplos

El técnico municipal de GIS desea visualizar los multipolígonos de zonas edificadas como puntos de un gráfico de densidad de edificación.

Las zonas edificadas se guardan en la tabla BUILDINGFOOTPRINTS que se creó con la siguiente sentencia CREATE TABLE.

```
CREATE TABLE BUILDINGFOOTPRINTS (building_id integer,
                                 lot_id      integer,
                                 footprint   db2gse.ST_MultiPolygon);
```

La función ST_Centroid devuelve el centro de cada multipolígono de zona edificada. La función AsShape convierte el punto central en una forma, que es la representación externa que la aplicación reconoce.

```
SELECT building_id,
       CAST(db2gse.AsShape(db2gse.ST_Centroid (footprint)) as blob(1m))
"Centroid"
FROM BUILDINGFOOTPRINTS;
```

ST_Contains

ST_Contains utiliza como entrada dos objetos de geometría y devuelve un 1 (TRUE) si el primer objeto contiene el segundo por completo y 0 (FALSE) si no es así.

Sintaxis

db2gse.ST_Contains(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

En el ejemplo siguiente se crean dos tablas. Una tabla contiene las zonas edificadas de la ciudad y la otra contiene sus parcelas. El ingeniero municipal desea asegurarse de que todas las áreas edificadas quedan completamente dentro de sus parcelas.

En ambas tablas, el tipo de datos multipolígono guarda la geometría de las zonas edificadas y de las parcelas. El diseñador de la base de datos ha seleccionado multipolígonos para ambos elementos geográficos. El diseñador ha observado que las parcelas pueden estar separadas por elementos geográficos naturales, tales como un río, y que las zonas edificadas pueden a menudo constar de varios edificios.

```
CREATE TABLE BUILDINGFOOTPRINTS (building_id integer,
                                 lot_id      integer,
                                 footprint   db2gse.ST_MultiPolygon);

CREATE TABLE LOTS (lot_id integer, lot db2gse.ST_MultiPolygon);
```

El ingeniero municipal selecciona primero los edificios que no quedan completamente dentro de una parcela.

```
SELECT building_id
FROM BUILDINGFOOTPRINTS, LOTS
WHERE db2gse.ST_Contains(lot,footprint) = 0;
```

El ingeniero municipal se da cuenta de que la primera consulta devolverá una lista de todos los ID de edificios cuyas áreas quedan fuera de un polígono de parcela. Pero el ingeniero municipal también sabe que esta información no indica si los otros edificios tienen asignado el ID de parcela correcto. Esta segunda consulta realiza una comprobación de integridad de los datos de la columna lot_id de la tabla BUILDINGFOOTPRINTS.

```
SELECT bf.building_id "building id", bf.lot_id "buildings lot_id",
       LOTS.lot_id "LOTS lot_id"
FROM BUILDINGFOOTPRINTS bf, LOTS
WHERE db2gse.ST_Contains(lot,footprint) = 1 AND LOTS.lot_id <> bf.lot_id;
```

En la Figura 30, las zonas edificadas marcadas con sus ID de edificio quedan dentro de sus parcelas. Las líneas de las parcelas se muestran con líneas de puntos. Aunque no se muestre, estas líneas alcanzan la línea central de calles y comprenden por completo las parcelas y las zonas edificadas que contienen.

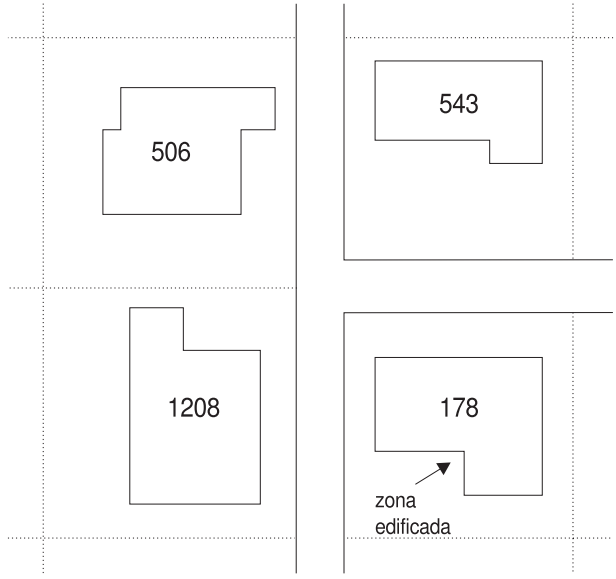


Figura 30. Utilización de ST_Contains para asegurar que todos los edificios quedan dentro de sus parcelas

ST_ConvexHull

ST_ConvexHull utiliza como entrada un objeto de geometría y devuelve el casco convexo.

Sintaxis

db2gse.ST_ConvexHull(g db2gse.ST_Geometry)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

El ejemplo crea la tabla CONVEXHULL_TEST que tiene dos columnas: GEOTYPE y G1. La columna GEOTYPE, tipo varchar(20), guardará el nombre de la subclase de geometría que se guarda en G1, que está definida como una geometría.

```
CREATE TABLE CONVEXHULL_TEST (geotype varchar(20), g1 db2gse.ST_Geometry)
```

Cada sentencia INSERT inserta una geometría de cada tipo de subclase en la tabla CONVEXHULL_TEST.

```
INSERT INTO CONVEXHULL_TEST
VALUES('Point',
       db2gse.ST_PointFromText('point (10.02 20.01)',
db2gse.coordref()..srid(0)))

INSERT INTO CONVEXHULL_TEST
VALUES('Linestring',
       db2gse.ST_LineFromText('linestring  (10.02 20.01,10.32 23.98,
       11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO CONVEXHULL_TEST
VALUES('Polygon',
       db2gse.ST_PolyFromText('polygon ((10.02 20.01,11.92 35.64,25.02 34.15,
                                         19.15 33.94,10.02 20.01))',
                    db2gse.coordref()..srid(0)))

INSERT INTO CONVEXHULL_TEST
VALUES('Multipoint',
       db2gse.ST_MPointFromText('multipoint (10.02 20.01,10.32 23.98,
       11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO CONVEXHULL_TEST
VALUES('Multilinestring',
       db2gse.ST_MLineFromText('multilinestring ((10.02 20.01,10.32 23.98,
              11.92 25.64),(9.55 23.75,15.36 30.11))',
                     db2gse.coordref()..srid(0)))

INSERT INTO CONVEXHULL_TEST
VALUES('Multipolygon',
       db2gse.ST_MPolyFromText('multipolygon (((10.02 20.01,11.92 35.64,
       25.02 34.15, 19.15 33.94,10.02 20.01)),
```

```
                                    ((51.71 21.73,73.36 27.04,71.52 32.87,
                                      52.43 31.90,51.71 21.73)))',
                        db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT lista el nombre de las subclases guardadas en la columna GEOTYPE y el casco convexo. El casco convexo generado por la función ST_ConvexHull se convierte en texto mediante la función ST_AsText. Luego se convierte a varchar(256) porque la salida por omisión de ST_AsText es varchar(4000).

```
SELECT GEOTYPE, CAST(db2gse.ST_AsText(db2gse.ST_ConvexHull(G1)))
as varchar(256) "The convexhull"
FROM CONVEXHULL_TEST
```

ST_CoordDim

ST_CoordDim devuelve las dimensiones de coordenadas del valor ST_Geometry. Para ver una explicación de las dimensiones de las coordenadas, consulte “Puntos” en la página 167.

Sintaxis

db2gse.ST_CoordDim(g1 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La tabla coorddim_test se crea con las columnas geotype y g1. La columna GEOTYPE guarda el nombre de la subclase de geometría guardada en la columna de geometría g1.

```
CREATE TABLE coorddim_test (geotype varchar(20), g1 db2gse.ST_Geometry)
```

Las sentencias INSERT insertan una subclase de ejemplo en la tabla coorddim_test.

```
INSERT INTO coorddim_test VALUES(
    'Point', db2gse.ST_PointFromText('point (10.02 20.01)',
db2gse.coordref()..srid(0))
)

INSERT INTO coorddim_test VALUES(
'Linestring',
       db2gse.ST_LineFromText('linestring  (10.02 20.01,10.32 23.98,11.92 25.64)',
db2gse.coordref()..srid(0))
)

INSERT INTO coorddim_test VALUES(
'Polygon', db2gse.ST_PolyFromText('polygon ((10.02 20.01,11.92 35.64,
25.02 34.15, 19.15 33.94,10.02 20.01))', db2gse.coordref()..srid(0))
)

INSERT INTO coorddim_test VALUES(
'Multipoint', db2gse.ST_MPointFromText('multipoint (10.02 20.01,10.32 23.98,
11.92 25.64)', db2gse.coordref()..srid(0))
)

INSERT INTO coorddim_test VALUES(
'Multilinestring', db2gse.ST_MLineFromText('multilinestring ((10.02 20.01,
10.32 23.98,11.92 25.64),(9.55 23.75,15.36 30.11))', db2gse.coordref()..srid(0))
)

INSERT INTO coorddim_test VALUES(
'Multipolygon',
MPolyFromText('multipolygon (((10.02 20.01,11.92 35.64,25.02 34.15,
19.15 33.94,10.02 20.01)),((51.71 21.73,73.36 27.04,71.52 32.87,
52.43 31.90,51.71 21.73)))', db2gse.coordref()..srid(0))
)
```

La sentencia SELECT lista el nombre de subclase guardado en la columna GEOTYPE (tipo de geometría) junto con la dimensión de coordenadas de ese tipo de geometría.

```
SELECT geotype, db2gse.ST_coordDim(g1)' coordinate_dimension'
FROM coorddim_test

GEOTYPE              coordinate_dimension
-------------------- --------------------------
ST_Point                                2
ST_Linestring                           2
ST_Polygon                              2
ST_Multipoint                           2
ST_Multilinestring                      2
ST_Multipolygon                         2

  6 record(s) selected.
```

ST_Crosses

ST_Crosses utiliza como entrada dos objetos de geometría y devuelve un 1 (TRUE) si su intersección da como resultado un objeto de geometría cuya dimensión es un nivel inferior a la dimensión máxima de los objetos fuente. El objeto intersección contiene puntos interiores a ambas geometrías de origen y no es igual a ninguno de los objetos fuente. Si no es así, devuelve un 0 (FALSE).

Sintaxis

db2gse.ST_Crosses(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

El gobierno regional está considerando la posibilidad de sacar una nueva ley según la cual los almacenes de residuos peligrosos de la región no pueden estar a menos de cinco millas de una vía fluvial. El gestor de GIS regional tiene una representación precisa de los ríos y corrientes, los cuales se guardan como multilíneas en la tabla WATERWAYS. Pero el gestor de GIS sólo tiene una ubicación de un solo punto para cada uno de los almacenes de residuos peligrosos.

```
CREATE TABLE WATERWAYS (id     integer,
                        name   varchar(128),
                        water  db2gse.ST_MultiLineString);

CREATE TABLE HAZARDOUS_SITES ( site_id   integer,
                               name      varchar(128),
                               location  db2gse.ST_Point);
```

Para determinar si el supervisor regional tiene que avisar a los almacenes que vulneran la ley propuesta, el gestor de GIS debe proteger las ubicaciones peligrosas y ver si algún río o corriente cruza el polígono de protección. El predicado ST_Crosses compara los sitios de la tabla HAZARDOUS_SITES rodeados con WATERWAYS. De este modo sólo se devuelven los registros en los que la vía fluvial cruza el radio de la región sobre el que se aplica la ley propuesta.

```
SELECT ww.name "River or stream", hs.name "Hazardous site"
FROM WATERWAYS ww, HAZARDOUS_SITES hs
WHERE db2gse.ST_Crosses(db2gse.ST_Buffer(hs.location,(5 * 5280)),ww.water)
=  1;
```

En la Figura 31 en la página 241, la zona de influencia de cinco millas de las zonas de residuos peligrosos cruza la red de corrientes que fluye por el distrito administrativo del condado. La red de corrientes se ha definido como

multilínea. Por tanto, el resultado incluye todos los segmentos de línea que forman parte de los segmentos que cruzan el radio.

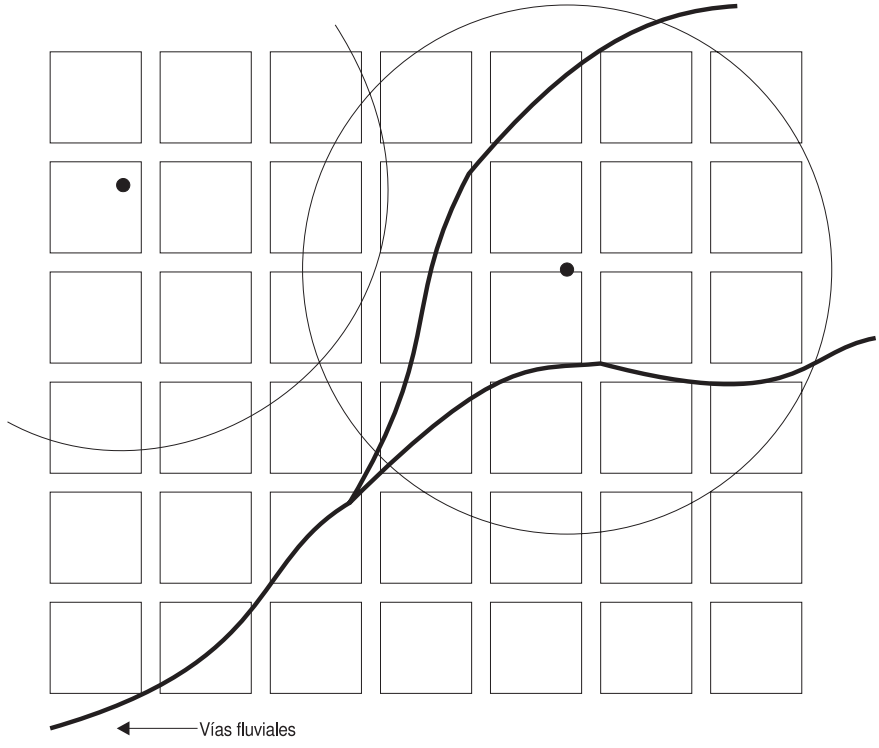


Figura 31. Utilización de ST_Crosses para encontrar las vías fluviales que pasan por una zona de residuos peligrosos

ST_Difference

ST_Difference utiliza como entrada dos objetos de geometría y devuelve un objeto de geometría que es la diferencia de los objetos fuente. Las dos geometrías de entrada de ST_Difference deben ser de la misma dimensión, de lo contrario el resultado es nulo.

Sintaxis

db2gse.ST_Difference(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

El ingeniero municipal desea saber el área total de las áreas de parcela de la ciudad que no están edificadas. Es decir, desea saber la suma de las áreas de parcelas sin contar las zonas edificadas.

```
CREATE TABLE BUILDINGFOOTPRINTS (building_id integer,
                                 lot_id      integer,
                                 footprint   db2gse.ST_MultiPolygon);

CREATE TABLE LOTS (lot_id   integer,
                   lot   db2gse.ST_MultiPolygon);
```

El ingeniero municipal une las tablas BUILDINGFOOTPRINTS y LOTS por la columna lot_id. Luego el ingeniero municipal toma la suma del área de la diferencia de las parcelas menos las zonas edificadas.

```
SELECT SUM(db2gse.ST_Area(db2gse.ST_Difference(lot,footprint)))
FROM BUILDINGFOOTPRINTS bf, LOTS
WHERE bf.lot_id = LOTS.lot_id;
```

ST_Dimension

ST_Dimension utiliza como entrada un objeto de geometría y devuelve su dimensión.

Sintaxis

db2gse.ST_Dimension(g1 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La tabla DIMENSION_TEST se crea con las columnas GEOTYPE y G1. La columna GEOTYPE guarda el nombre de la subclase de geometría que está almacenada en la columna de geometría G1.

```
CREATE TABLE DIMENSION_TEST (geotype varchar(20), g1 db2gse.ST_Geometry)
```

Las sentencias INSERT insertan una subclase de ejemplo en la tabla DIMENSION_TEST.

```
INSERT INTO DIMENSION_TEST
VALUES('Point',
       db2gse.ST_PointFromText('point (10.02 20.01)',
                                    db2gse.coordref()..srid(0)))

INSERT INTO DIMENSION_TEST
VALUES('Linestring',
       db2gse.ST_LineFromText('linestring  (10.02 20.01,10.32 23.98,
               11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO DIMENSION_TEST
VALUES('Polygon',
       db2gse.ST_PolyFromText('polygon ((10.02 20.01,11.92 35.64,25.02 34.15,
                                         19.15 33.94,10.02 20.01))',
                     db2gse.coordref()..srid(0)))

INSERT INTO DIMENSION_TEST
VALUES('Multipoint',
       db2gse.ST_MPointFromText('multipoint (10.02 20.01,10.32 23.98,
               11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO DIMENSION_TEST
VALUES('Multilinestring',
       db2gse.ST_MLineFromText('multilinestring ((10.02 20.01,10.32 23.98,
              11.92 25.64),(9.55 23.75,15.36 30.11))',
                     db2gse.coordref()..srid(0)))

INSERT INTO DIMENSION_TEST
VALUES('Multipolygon',
       db2gse.ST_MPolyFromText('multipolygon (((10.02 20.01,11.92 35.64,
              25.02 34.15,19.15 33.94,10.02 20.01)),
```

```
((51.71 21.73,73.36 27.04,71.52 32.87,
52.43 31.90,51.71 21.73)))',
    db2gse.coordref)..srid(0)))
```

La siguiente sentencia SELECT lista el nombre de subclase almacenado en la columna GEOTYPE (tipo de geometría) junto con la dimensión de ese tipo de geometría.

```
SELECT geotype, db2gse.ST_Dimension(g1) "The dimension"
FROM DIMENSION_TEST
```

Se devuelve el siguiente conjunto resultante.

```
GEOTYPE              The dimension
-------------------- -------------
ST_Point                        0
ST_Linestring                   1
ST_Polygon                      2
ST_Multipoint                   0
ST_Multilinestring              1
ST_Multipolygon                 2

  6 record(s) selected.
```

ST_Disjoint

ST_Disjoint utiliza como entrada dos geometrías y devuelve un 1 (TRUE) si la intersección de las dos geometrías da como resultado un conjunto vacío y 0 (FALSE) si no es así.

Sintaxis

db2gse.ST_Disjoint(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

Una compañía de seguros desea evaluar la cobertura de seguro del hospital, las guarderías y los colegios de una ciudad. Parte de este proceso incluye determinar la amenaza que suponen los sitios de residuos peligrosos para cada institución. La compañía de seguros desea tener en cuenta las instituciones que no están expuestas a la contaminación. El consultor de GIS contratado por la compañía de seguros debe localizar todas las instituciones que no se encuentran en un radio de cinco millas de un sitio de residuos peligrosos.

La tabla SENSITIVE_AREAS contiene varias columnas que describen las instituciones amenazadas además de la columna ZONE, que guarda la geometría de polígono de la institución.

```
CREATE TABLE SENSITIVE_AREAS (id        integer,
                              name      varchar(128),
                              size      float,
                              type      varchar(10),
                              zone      db2gse.ST_Polygon);
```

La tabla HAZARDOUS_SITES contiene el identificador de los sitios en las columnas SITE_ID y NAME, mientras que la ubicación geográfica real de cada sitio se guarda en la columna LOCATION.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(128),
                              location  db2gse.ST_Point);
```

La siguiente sentencia SELECT lista los nombres de todas las áreas sensibles que no se encuentran en un radio de 5 millas de un sitio de residuos peligrosos. La función ST_Intersects podría sustituir a la función ST_Disjoint en esta consulta si el resultado de la función se define como igual a 0 en lugar de 1. Esto se debe a que ST_Intersects y ST_Disjoint devuelven resultados exactamente opuestos.

```
SELECT sa.name
FROM SENSITIVE_AREAS sa, HAZARDOUS_SITES hs
WHERE db2gse.ST_Disjoint(db2gse.ST_Buffer(hs.location,(5 * 5280)),sa.zone) = 1;
```

En la Figura 32, se comparan las áreas sensibles con el radio de cinco millas de los sitios de residuos peligrosos. La guardería es la única área sensible en la que la función ST_Disjoint devuelve un 1 (TRUE). La función ST_Disjoint devuelve un 1 cuando dos geometrías no forman ninguna intersección.

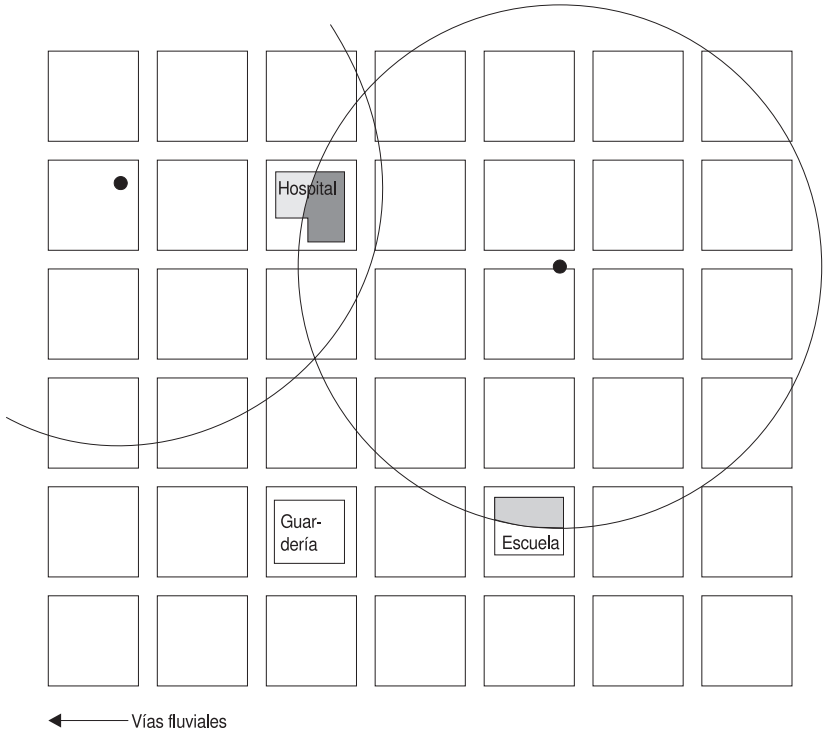


Figura 32. Utilización de ST_Disjoint para buscar edificios que no quedan dentro de ningún área de residuos peligrosos (no forman intersección con la misma)

ST_Distance

ST_Distance utiliza como entrada dos geometrías y devuelve la distancia mínima que las separa.

Sintaxis

db2gse.ST_Distance(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Double

Ejemplos

El ingeniero municipal necesita una lista de todos los edificios que están a un pie o menos de una línea de parcela.

La columna BUILDING_ID de la tabla BUILDINGFOOTPRINTS identifica de forma exclusiva cada edificio. La columna LOT_ID identifica la parcela a la que pertenece cada edificio. El multipolígono de zona ocupada guarda la zona ocupada correspondiente a cada edificio.

```
CREATE TABLE BUILDINGFOOTPRINTS (  building_id integer,
                                   lot_id      integer,
                                   footprint   db2gse.ST_MultiPolygon);
```

La tabla LOTS guarda el ID de parcela que identifica de forma exclusiva a cada parcela y el multipolígono de parcela que contiene la geometría de línea de la parcela.

```
CREATE TABLE LOTS ( lot_id   integer,
                    lot      db2gse.ST_MultiPolygon);
```

La consulta devuelve una lista de los ID de edificios que están a un pie o menos de sus líneas de parcela. La función ST_Distance realiza una unión espacial entre los multipolígonos correspondientes a zonas edificadas y al límite de la parcela. Sin embargo, la unión entre bf.lot_id y LOTS.lot_id asegura que la función ST_Distance sólo compara los multipolígonos que pertenecen a la misma parcela.

```
SELECT bf.building_id
  FROM BUILDINGFOOTPRINTS bf, LOTS
 WHERE bf.lot_id = LOTS.lot_id AND
       db2gse.ST_Distance(bf.footprint, db2gse.ST_Boundary(LOTS.lot)) <= 1.0;
```

ST_Endpoint

ST_Endpoint utiliza como entrada una línea y devuelve un punto que es el último punto de la línea.

Sintaxis

db2gse.ST_Endpoint(c db2gse.ST_Curve)

Tipo devuelto

db2gse.ST_Point

Ejemplos

La tabla ENDPOINT_TEST guarda la columna de enteros GID que identifica de forma exclusiva cada fila y la columna de líneas LN1 que guarda líneas.

```
CREATE TABLE ENDPOINT_TEST (gid integer, ln1 db2gse.ST_LineString)
```

Las sentencias INSERT insertan líneas en la tabla ENDPOINT_TEST. La primera no tiene coordenadas Z ni medidas; la segunda, sí.

```
INSERT INTO ENDPOINT_TEST
VALUES( 1,
        db2gse.ST_LineFromText('linestring (10.02 20.01,23.73 21.92,
        30.10 40.23)', db2gse.coordref()..srid(0)))

INSERT INTO ENDPOINT_TEST
VALUES(2,
       db2gse.ST_LineFromText('linestring zm (10.02 20.01 5.0 7.0,
       23.73 21.92 6.5 7.1, 30.10 40.23 6.9 7.2)',
                    db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT lista la columna GID con la salida de la función ST_Endpoint. La función ST_Endpoint genera una geometría punto que la función ST_AsText convierte en texto. Se utiliza la función CAST para abreviar el valor varchar(4000) por omisión de la función ST_AsText a un valor varchar(60).

```
SELECT gid, CAST(db2gse.ST_AsText(db2gse.ST_Endpoint(ln1)) AS varchar(60))
"Endpoint"
FROM ENDPOINT_TEST
```

Se devuelve el siguiente conjunto resultante.

```
GID         Endpoint
----------- ------------------------------------------------------------
          1 POINT ( 30.10000000 40.23000000)
          2 POINT ZM ( 30.10000000 40.23000000 7.00000000 7.20000000)

  2 record(s) selected.
```

ST_Envelope

ST_Envelope utiliza como entrada un objeto de geometría y devuelve el recuadro que lo limita como una geometría.

Sintaxis

db2gse.ST_Envelope(g db2gse.ST_Geometry)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

La columna GEOTYPE de la tabla ENVELOPE_TEST guarda el nombre de la subclase de geometría guardada en la columna de geometría G1.

```
CREATE TABLE ENVELOPE_TEST (geotype varchar(20), g1 db2gse.ST_Geometry)
```

Las siguientes sentencias INSERT insertan cada subclase de geometría en la tabla ENVELOPE_TEST.

```
INSERT INTO ENVELOPE_TEST
VALUES('Point',
       db2gse.ST_PointFromText('point (10.02 20.01)',
                                db2gse.coordref()..srid(0)))

INSERT INTO ENVELOPE_TEST
VALUES ('Linestring',
        db2gse.ST_LineFromText('linestring (10.01 20.01, 10.01 30.01,
        10.01 40.01)', db2gse.coordref()..srid(0)))

INSERT INTO ENVELOPE_TEST
VALUES('Linestring',
       db2gse.ST_LineFromText('linestring  (10.02 20.01,10.32 23.98,11.92 25.64)',
       db2gse.coordref()..srid(0)))

INSERT INTO ENVELOPE_TEST
VALUES('Polygon',
       db2gse.ST_PolyFromText('polygon ((10.02 20.01,11.92 35.64,25.02 34.15,
                                         19.15 33.94,10.02 20.01))',
                    db2gse.coordref()..srid(0)))

INSERT INTO ENVELOPE_TEST
VALUES('Multipoint',
       db2gse.ST_MPointFromText('multipoint (10.02 20.01,10.32 23.98,11.92 25.64)',
       db2gse.coordref()..srid(0)))

INSERT INTO ENVELOPE_TEST
VALUES('Multilinestring',
       db2gse.ST_MLineFromText('multilinestring ((10.01 20.01,20.01 20.01,
       30.01 20.01), (30.01 20.01,40.01 20.01,50.01 20.01))',
                     db2gse.coordref()..srid(0)))

INSERT INTO ENVELOPE_TEST
VALUES('Multilinestring',
       db2gse.ST_MLineFromText('multilinestring ((10.02 20.01,10.32 23.98,
       11.92 25.64), ( 9.55 23.75,15.36 30.11))',
                     db2gse.coordref()..srid(0)))

INSERT INTO ENVELOPE_TEST
VALUES('Multipolygon',
       db2gse.ST_MPolyFromText('multipolygon (((10.02 20.01,11.92 35.64,
```

```
      25.02 34.15, 19.15 33.94,10.02 20.01)),
                                    ((51.71 21.73,73.36 27.04,71.52 32.87,
                                      52.43 31.90,51.71 21.73)))',
                  db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT lista el nombre de subclase junto a su envolvente. Puesto que la función ST_Envelope devuelve un punto, una línea o un polígono, su salida se convierte en texto con la función ST_AsText. La función CAST convierte el resultado varchar(4000) por omisión de la función ST_AsText a un valor varchar(280).

```
SELECT GEOTYPE, CAST(db2gse.ST_AsText(db2gse.ST_Envelope(g1)) AS varchar(280))
"The envelope"
FROM ENVELOPE_TEST
```

Se devuelve el siguiente conjunto resultante.

```
GEOTYPE              The envelope
-------------------- ------------------------------------------------------------
Point                POINT ( 10.02000000 20.01000000)
Linestring           LINESTRING ( 10.01000000 20.01000000, 10.01000000
 40.01000000)
Linestring           POLYGON (( 10.02000000 20.01000000, 11.92000000
 20.01000000, 11.92000000 25.64000000, 10.02000000 25.64000000, 10.02000000
 20.01000000))
Polygon              POLYGON (( 10.02000000 20.01000000, 25.02000000
 20.01000000, 25.02000000 35.64000000, 10.02000000 35.64000000, 10.02000000
 20.01000000))
Multipoint           POLYGON (( 10.02000000 20.01000000, 11.92000000
 20.01000000, 11.92000000 25.64000000, 10.02000000 25.64000000, 10.02000000
 20.01000000))
Multilinestring      LINESTRING ( 10.01000000 20.01000000, 50.01000000
 20.01000000)
Multilinestring      POLYGON (( 9.55000000 20.01000000, 15.36000000
 20.01000000, 15.36000000 30.11000000, 9.55000000 30.11000000, 9.55000000
 20.01000000))
Multipolygon         POLYGON (( 10.02000000 20.01000000, 73.36000000
 20.01000000, 73.36000000 35.64000000, 10.02000000 35.64000000, 10.02000000
 20.01000000))

  8 record(s) selected.
```

ST_Equals

ST_Equals compara dos geometrías y devuelve un 1 (TRUE) si las geometrías son idénticas y 0 (FALSE) si no es así.

Sintaxis

db2gse.ST_Equals(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

El técnico de GIS sospecha que algunos datos de la tabla BUILDINGFOOTPRINTS están duplicados. El técnico consulta la tabla para determinar si cualquiera de los multipolígonos de zona edificada son iguales.

La tabla BUILDINGFOOTPRINTS se crea con la siguiente sentencia. La columna BUILDING_ID identifica de forma exclusiva los edificios, la columna LOT_ID identifica la parcela del edificio y la columna FOOTPRINT guarda la geometría del edificio.

```
CREATE TABLE BUILDINGFOOTPRINTS ( building_id integer,
                                  lot_id      integer,
                                  footprint   db2gse.ST_MultiPolygon);
```

La tabla BUILDINGFOOTPRINTS se une espacialmente a sí misma mediante el predicado ST_Equals, el cual devuelve un 1 si encuentra dos multipolígonos iguales. Se necesita la condición bf1.building_id <> bf2.building_id para eliminar la comparación de la misma geometría.

```
SELECT bf1.building_id, bf2.building_id
FROM BUILDINGFOOTPRINTS bf1, BUILDINGFOOTPRINTS bf2
WHERE db2gse.ST_Equals(bf1.footprint,bf2.footprint) = 1
      and bf1.building_id <> bf2.building_id;
```

ST_ExteriorRing

ST_ExteriorRing utiliza como entrada un polígono y devuelve su anillo exterior como una línea.

Sintaxis

db2gse.ST_ExteriorRing(s db2gse.ST_Polygon)

Tipo devuelto

db2gse.ST_LineString

Ejemplos

Un ornitólogo que está estudiando la población de aves de varias islas de los mares del sur sabe que la zona en la que se alimenta una determinada especie de aves está restringida a la orilla. Para calcular la capacidad de las islas para alimentar a esta especie, el ornitólogo necesita saber el perímetro de las islas. Aunque algunas de las islas contienen estanques, las orillas de estos estanques están habitadas exclusivamente por otras especies de aves más agresivas. Por lo tanto, el ornitólogo necesita saber el perímetro del anillo exterior de las islas.

Las columnas ID y NAME de la tabla ISLANDS identifican cada isla y la columna LAND de tipo ST_Polygon guarda la geometría de cada una de ellas.

```
CREATE TABLE ISLANDS (id   integer,
                      name   varchar(32),
                      land   db2gse.ST_Polygon);
```

La función ST_ExteriorRing extrae el anillo exterior del polígono de cada isla como una línea. La longitud de la línea se establece mediante la función length. Las longitudes de las líneas se suman mediante la función SUM.

```
SELECT SUM(db2gse.ST_length(db2gse.ST_ExteriorRing (land))) FROM ISLANDS;
```

En la Figura 33 en la página 253, los anillos exteriores de las islas representan la interfaz ecológica que cada isla comparte con el mar. Algunas de las islas tienen lagos, que se representan mediante los anillos interiores de los polígonos.

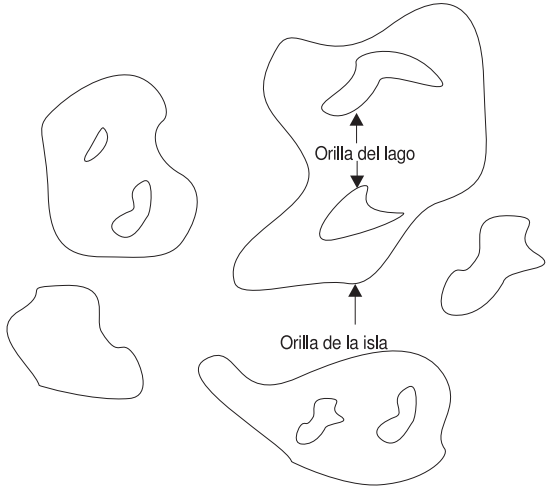


Figura 33. Utilización de ST_ExteriorRing para determinar la longitud de la orilla de una isla

ST_GeometryN

ST_GeometryN utiliza como entrada un grupo y un índice entero y devuelve el objeto de geometría número *n* del grupo.

Sintaxis

db2gse.ST GeometryN(g db2gse.ST_Geometry, n Integer)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

El ingeniero municipal desea saber si las zonas edificadas están todas dentro del primer polígono del multipolígono de la parcela.

La columna BUILDING_ID identifica de forma exclusiva cada fila de la tabla BUILDINGFOOTPRINTS. La columna LOT_ID identifica la parcela del edificio. La columna FOOTPRINT guarda la geometría del edificio.

```
CREATE TABLE BUILDINGFOOTPRINTS ( building_id integer,
                                  lot_id      integer,
                                  footprint   db2gse.ST_MultiPolygon);

CREATE TABLE LOTS ( lot_id   integer,
                    lot   db2gse.ST_MultiPolygon);
```

La consulta lista las columnas building_id y lot_id de la tabla BUILDINGFOOTPRINTS correspondientes a las zonas edificadas que quedan dentro del primer polígono de parcela. La función ST_GeometryN devuelve un primer polígono de parcela de la matriz de multipolígono.

```
SELECT bf.building_id,bf.lot_id
FROM BUILDINGFOOTPRINTS bf,LOTS
WHERE db2gse.ST_Within(footprint, db2gse.ST_GeometryN (lot,1)) = 1
      AND bf.lot_id = LOTS.lot_id;
```

ST_GeometryType

ST_GeometryType utiliza como entrada un objeto ST_Geometry y devuelve su tipo de geometría como una serie.

Sintaxis

db2gse.ST_GeometryType (g db2gse.ST_Geometry)

Tipo devuelto

Varchar(4000)

Ejemplos

La tabla GEOMETRYTYPE_TEST contiene la columna de geometría G1.

```
CREATE TABLE GEOMETRYTYPE_TEST(g1 db2gse.ST_Geometry)
```

Las siguientes sentencias INSERT insertan cada subclase de geometría en la columna G1.

```
INSERT INTO GEOMETRYTYPE_TEST
VALUES(db2gse.ST_GeomFromText('point (10.02 20.01)',
       db2gse.coordref()..srid(0)))

INSERT INTO GEOMETRYTYPE_TEST
VALUES (db2gse.ST_GeomFromText('linestring (10.01 20.01, 10.01 30.01,
        10.01 40.01)', db2gse.coordref()..srid(0)))

INSERT INTO GEOMETRYTYPE_TEST
VALUES(db2gse.ST_Geometrytype_test values(db2gse.ST_GeomFromText('polygon
      ((10.02 20.01,11.92 35.64,25.02 34.15,19.15 33.94, 10.02 20.01))',
                        db2gse.coordref()..srid(0))))

INSERT INTO GEOMETRYTYPE_TEST
VALUES(db2gse.ST_GeomFromText('multipoint (10.02
 20.01,10.32 23.98,
       11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO GEOMETRYTYPE_TEST
VALUES(db2gse.ST_GeomFromText('multilinestring ((10.02 20.01,10.32 23.98,
                                          11.92 25.64),
                                        (9.55 23.75,15.36 30.11))',
                        db2gse.coordref()..srid(0)))

INSERT INTO GEOMETRYTYPE_TEST
VALUES(db2gse.ST_GeomFromText('multipolygon (((10.02 20.01,11.92 35.64,
        25.02 34.15, 19.15 33.94,10.02 20.01)),
                                      ((51.71 21.73,73.36 27.04,71.52 32.87,
                                        52.43 31.90,51.71 21.73)))',
                        db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT lista el tipo de geometría de cada subclase que se guarda en la columna de geometría G1.

```
SELECT db2gse.ST_GeometryType(g1) "Geometry type" FROM GEOMETRYTYPE_TEST
```

Se devuelve el siguiente conjunto resultante.

```
Geometry type
------------------------------
ST_Point
ST_LineString
ST_Polygon
ST_MultiPoint
ST_MultiLineString
ST_MultiPolygon

  6 record(s) selected.
```

ST_GeomFromText

ST_GeomFromText utiliza como entrada una representación de texto convencional y un identificador de sistema de referencias espaciales, y devuelve un objeto de geometría.

Sintaxis

db2gse.ST_GeomFromText(geometryTaggedText Varchar(4000), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

La tabla GEOMETRY_TEST contiene la columna de enteros GID, que identifica de forma exclusiva cada fila, y la columna G1, que guarda la geometría.

```
CREATE TABLE GEOMETRY_TEST (gid smallint, g1 db2gse.ST_Geometry)
```

Las sentencias INSERT insertan los datos en las columnas GID y G1 de la tabla GEOMETRY_TEST. La función ST_GeomFromText convierte la representación de texto de cada geometría en su correspondiente subclase replicable de Spatial Extender.

```
INSERT INTO GEOMETRY_TEST
VALUES(1, db2gse.ST_GeomFromText('point (10.02 20.01)',
     db2gse.coordref()..srid(0)))

INSERT INTO GEOMETRY_TEST
VALUES (2,
        db2gse.ST_GeomFromText('linestring (10.01 20.01, 10.01 30.01,
        10.01 40.01)', db2gse.coordref()..srid(0)))

INSERT INTO GEOMETRY_TEST
VALUES(3,
       db2gse.ST_GeomFromText('polygon ((10.02 20.01,11.92 35.64,25.02 34.15,
                                         19.15 33.94,10.02 20.01))',
                        db2gse.coordref()..srid(0)))

INSERT INTO GEOMETRY_TEST
VALUES(4,
       db2gse.ST_GeomFromText('multipoint (10.02 20.01,10.32 23.98,
       11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO GEOMETRY_TEST
VALUES(5,
       db2gse.ST_GeomFromText('multilinestring ((10.02 20.01,10.32 23.98,
                                          11.92 25.64),
                                         ( 9.55 23.75,15.36 30.11))',
                        db2gse.coordref()..srid(0)))
```

```
INSERT INTO GEOMETRY_TEST
VALUES(6,
       db2gse.ST_GeomFromText('multipolygon (((10.02 20.01,11.92 35.64,
      25.02 34.15, 19.15 33.94,10.02 20.01)),
                                        ((51.71 21.73,73.36 27.04,71.52 32.87,
                                          52.43 31.90,51.71 21.73)))',
                         db2gse.coordref()..srid(0)))
```

ST_GeomFromWKB

ST_GeomFromWKB utiliza como entrada una representación binaria convencional y un identificador de sistema de referencias espaciales, y devuelve un objeto de geometría.

Sintaxis

db2gse.ST_GeomFromWKB(WKBGeometry Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

El siguiente fragmento de código C contiene funciones ODBC intercaladas con funciones SQL de Spatial Extender que insertan datos en la tabla LOTS.

La tabla LOTS se creó con dos columnas: la columna LOT_ID, que identifica de forma exclusiva cada parcela, y la columna de multipolígono LOT, que contiene la geometría de cada parcela.

```
CREATE TABLE LOTS ( lot_id   integer,
                    lot      db2gse.ST_MultiPolygon);
```

La función ST_GeomFromWKB convierte las representaciones WKB en geometría de Spatial Extender. La sentencia INSERT completa se copia en una serie de caracteres wkb_sql. La sentencia INSERT contiene marcadores de parámetro para aceptar los datos LOT_ID y los datos LOT de forma dinámica.

```
/* Crear la sentencia insert de SQL para entrar datos
en el ID de parcela y el multipolígono de parcela. Los interrogantes son
marcadores de parámetro que indican el ID de parcela y los valores de
parcela que se recuperarán durante la ejecución. */
strcpy (wkb_sql,"insert into LOTS (lot_id, lot) values (?,
  db2gse.ST_GeomFromWKB

(cast(? as blob(1m)), db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)wkb_sql, SQL_NTS);

/* Vincular el valor de clave de entero con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_SLONG,
     SQL_INTEGER, 0, 0, &lot_id, 0, &pcbvalue1);

/* Vincular la forma con el segundo parámetro. */
pcbvalue2 = blob_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_BINARY,
```

```
     SQL_BLOB, blob_len, 0, shape_blob, blob_len, &pcbvalue2);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

ST_InteriorRingN

Devuelve el anillo interior número *n* de un polígono como una línea. Los anillos no se organizan por orientación geométrica. Se organizan según las reglas definidas por las rutinas internas de verificación de geometrías. Así pues, el orden de los anillos no se puede predefinir.

Sintaxis

ST_InteriorRingN(p ST_Polygon, n Integer)

Tipo devuelto

db2gse.ST_LineString

Ejemplos

Un ornitólogo que está estudiando la población de aves de varias islas de los mares del sur sabe que la zona en la que se alimenta una determinada especie pasiva de aves está restringida a la orilla. Algunas de las islas contienen varios lagos. Las orillas de los lagos están habitadas exclusivamente por especies de aves más agresivas. El ornitólogo sabe que, en cada isla, si el perímetro de los lagos supera un determinado umbral, las especies agresivas serán tan numerosas que amenazarán a las especies pasivas de la orilla. Por lo tanto, el ornitólogo necesita saber la suma de perímetros de los anillos interiores de las islas.

En la Figura 34 en la página 262, los anillos exteriores de las islas representan la interfaz ecológica que cada isla comparte con el mar. Algunas de las islas tienen lagos, que se representan mediante los anillos interiores de los polígonos.

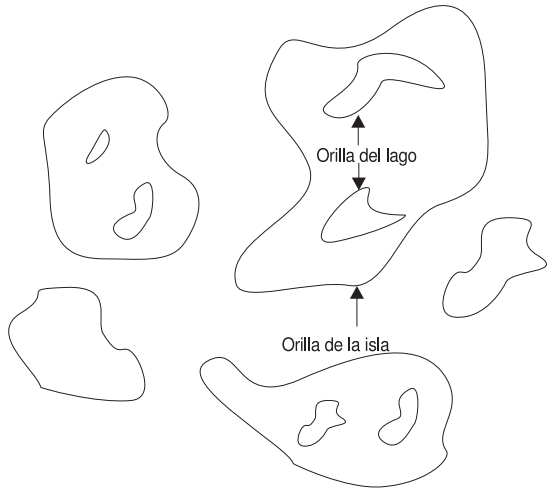


Figura 34. Utilización de ST_InteriorRingN para determinar la longitud de las orillas dentro de cada isla

Las columnas ID y name de la tabla ISLANDS identifican cada isla, mientras que la columna de polígonos land guarda la geometría de la isla.

```
CREATE TABLE ISLANDS (id    integer,
                      name  varchar(32),
                      land  db2gse.ST_Polygon);
```

El siguiente programa ODBC utiliza la función ST_InteriorRingN para extraer el anillo interior (lago) de cada polígono de islas como una línea. El perímetro de la línea que devuelve la función length se suma y se muestra junto con el ID de la isla.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

#include "sg.h"
#include "sgerr.h"
#include "sqlcli1.h"

/***                          ***
 *** Cambiar estas constantes ***
 ***                          ***/

#define USER_NAME   "sdetest"   /* su nombre de usuario */
#define USER_PASS   "acid.rain" /* su contraseña de usuario */
#define DB_NAME     "mydb"      /* base de datos a la que se va a conectar */

static void check_sql_err (SQLHDBC  handle,
                           SQLHSTMT hstmt,
                           LONG     rc,
                           CHAR     *str);

void main( argc, argv )
```

```
int argc;
char *argv[];
{
  SQLHDBC       handle;
  SQLHENV       henv;
  CHAR          sql_stmt[256];
  LONG          rc,
                total_perimeter,
                num_lakes,
                lake_number,
                island_id,
                lake_perimeter;
  SQLHSTMT      island_cursor,
                lake_cursor;
  SDWORD        pcbvalue,
                id_ind,
                lake_ind,
                length_ind;

/* Asignar memoria para el descriptor de entorno ODBC henv
e inicializar la aplicación. */

  rc = SQLAllocEnv (&henv);
  if (rc != SQL_SUCCESS)
  {
    printf ("SQLAllocEnv failed with %d\n", rc);
    exit(0);
  }

/* Asignar memoria para un descriptor de conexión del
entorno henv. */
  rc = SQLAllocConnect (henv, &handle);
  if (rc != SQL_SUCCESS)
  {
    printf ("SQLAllocConnect failed with %d\n", rc);
    exit(0);
  }

/* Cargar el controlador ODBC y conectar con la fuente de datos identificada
   por la base de datos, usuario y contraseña. */

  rc = SQLConnect (handle,
                   (UCHAR *)DB_NAME,
                   SQL_NTS,
                   (UCHAR *)USER_NAME,
                   SQL_NTS,
                   (UCHAR *)USER_PASS,
                   SQL_NTS);

  check_sql_err (handle, NULL, rc, "SQLConnect");

/* Asignar memoria al descriptor de sentencia SQL
island_cursor. */

  rc = SQLAllocStmt (handle, &island_cursor);
  check_sql_err (handle, NULL, rc, "SQLAllocStmt");

/* Preparar y ejecutar la consulta para obtener los ID de isla y el
   número de lagos (anillos interiores) */

  strcpy (sql_stmt, "select id, db2gse.ST_NumInteriorRings(land) from ISLANDS");

  rc = SQLExecDirect (island_cursor, (UCHAR *)sql_stmt, SQL_NTS);
  check_sql_err (NULL, island_cursor, rc, "SQLExecDirect");

/* Vincular la columna ID de la tabla con la variable island_id */
```

```
  rc = SQLBindCol (island_cursor, 1, SQL_C_SLONG, &island_id, 0, &id_ind);
  check_sql_err (NULL, island_cursor, rc, "SQLBindCol");

/* Vincular el resultado de numinteriorrings(land) con la variable num_lakes. */

  rc = SQLBindCol (island_cursor, 2, SQL_C_SLONG, &num_lakes, 0, &lake_ind);
  check_sql_err (NULL, island_cursor, rc, "SQLBindCol");

/* Asignar memoria al descriptor de sentencia SQL
lake_cursor. */

rc = SQLAllocStmt (handle, &lake_cursor);
  check_sql_err (handle, NULL, rc, "SQLAllocStmt");

/* Preparar la consulta para obtener la longitud de un anillo interior. */

  strcpy (sql_stmt,
          "select Length(db2gse.ST_InteriorRingN(land, cast (? as
           integer))) from ISLANDS where id = ?");

  rc = SQLPrepare (lake_cursor, (UCHAR *)sql_stmt, SQL_NTS);
  check_sql_err (NULL, lake_cursor, rc, "SQLPrepare");

/* Vincular la variable lake_number como el primer parámetro de entrada */

  pcbvalue = 0;
  rc = SQLBindParameter (lake_cursor, 1, SQL_PARAM_INPUT, SQL_C_LONG,
       SQL_INTEGER, 0, 0, &lake_number, 0, &pcbvalue);
  check_sql_err (NULL, lake_cursor, rc, "SQLBindParameter");

/* Vincular island_id como el segundo parámetro de entrada */

  pcbvalue = 0;
  rc = SQLBindParameter (lake_cursor, 2, SQL_PARAM_INPUT, SQL_C_LONG,
       SQL_INTEGER, 0, 0, &island_id, 0, &pcbvalue);
  check_sql_err (NULL, lake_cursor, rc, "SQLBindParameter");

/* Vincular el resultado de Length(db2gse.ST_InteriorRingN(land, cast
  (? as integer))) a la variable lake_perimeter */

  rc = SQLBindCol (lake_cursor, 1, SQL_C_SLONG, &lake_perimeter, 0,
                   &length_ind);
  check_sql_err (NULL, island_cursor, rc, "SQLBindCol");

  /* Bucle externo, obtener id de islas y el número de lagos
  (anillos interiores) */

  while (SQL_SUCCESS == rc)
  {
    /* Buscar y cargar una isla */

    rc = SQLFetch (island_cursor);

    if (rc != SQL_NO_DATA)
    {
      check_sql_err (NULL, island_cursor, rc, "SQLFetch");

      /* Bucle interno, para esta isla obtener el perímetro de todos
         sus lagos (anillos interiores) */

      for (total_perimeter = 0,lake_number = 1;
           lake_number <= num_lakes;
           lake_number++)
      {
        rc = SQLExecute (lake_cursor);
```

```
        check_sql_err (NULL, lake_cursor, rc, "SQLExecute");

        rc = SQLFetch (lake_cursor);
        check_sql_err (NULL, lake_cursor, rc, "SQLFetch");

        total_perimeter += lake_perimeter;

        SQLFreeStmt (lake_cursor, SQL_CLOSE);
      }
    }

/* Mostrar el id de isla y el perímetro total de sus lagos. */

    printf ("Island ID = %d,  Total lake perimeter = %d\n",
             island_id,total_perimeter);

  }

  SQLFreeStmt (lake_cursor, SQL_DROP);
  SQLFreeStmt (island_cursor, SQL_DROP);
  SQLDisconnect (handle);
  SQLFreeConnect (handle);
  SQLFreeEnv (henv);

  printf( "\nTest Complete ...\n" );

}

static void check_sql_err (SQLHDBC handle, SQLHSTMT  hstmt, LONG rc,
                           CHAR *str)
{

    SDWORD dbms_err = 0;
    SWORD  length;
    UCHAR   err_msg[SQL_MAX_MESSAGE_LENGTH], state[6];

    if (rc != SQL_SUCCESS)
    {
      SQLError (SQL_NULL_HENV, handle, hstmt, state, &dbms_err,
                err_msg, SQL_MAX_MESSAGE_LENGTH - 1, &length);
      printf ("%s ERROR (%d): DBMS code:%d, SQL state: %s, message:
               \n %s\n", str, rc, dbms_err, state, err_msg);

      if (handle)
      {
        SQLDisconnect (handle);
        SQLFreeConnect (handle);
      }
      exit(1);
    }
}
```

ST_Intersection

ST_Intersection utiliza como entrada dos objetos de geometría y devuelve el conjunto de intersecciones como un objeto de geometría.

Si ST_Intersection utiliza un polígono y una línea como parámetros de entrada, y se cumplen estas condiciones:

- Un punto del polígono es el punto de inicio de la línea, y
- El polígono y la línea no tienen otros puntos en común,

entonces ST_Intersection devuelve la cadena de texto POINT EMPTY.

Sintaxis

db2gse.ST_Intersection(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

El jefe de bomberos debe obtener las áreas de los hospitales, colegios y guarderías que forman intersección con el radio de una posible área de contaminación por residuos peligrosos.

Las áreas sensibles se guardan en la tabla SENSITIVE_AREAS que se crea con la siguiente sentencia CREATE TABLE. La columna ZONE está definida como un polígono que guarda el contorno de cada una de las áreas sensibles.

```
CREATE TABLE SENSITIVE_AREAS (id    integer,
                              name  varchar(128),
                              size  float,
                              type  varchar(10),
                              zone  db2gse.ST_Polygon);
```

Los sitios peligrosos se guardan en la tabla HAZARDOUS_SITES que se crea con la siguiente sentencia CREATE TABLE. La columna LOCATION, definida como un punto, almacena la ubicación que constituye el centro geográfico de cada sitio peligroso.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(128),
                              location  db2gse.ST_Point);
```

La función buffer genera una zona de separación de cinco millas que rodea a las zonas de residuos peligrosos. La función ST_Intersection genera polígonos a partir de la intersección de los polígonos de sitios de residuos peligrosos de la zona de influencia y las áreas sensibles. La función ST_Area devuelve el área de los polígonos de intersección, que se suma para cada sitio peligroso mediante la función SUM. La cláusula GROUP BY indica a la consulta que sume las áreas de la intersección por id del sitio de residuos peligrosos.

```
SELECT hs.name,SUM(db2gse.ST_Area(db2gse.ST_Intersection (sa.zone,
db2gse.ST_buffer hs.location,(5 * 5280))))
FROM SENSITIVE_AREAS sa, HAZARDOUS_SITES hs
GROUP BY hs.site_id;
```

En la Figura 35, los círculos representan los polígonos con zona de separación que hay alrededor de los sitios de residuos peligrosos. La intersección de estos polígonos con zona de influencia y los polígonos de áreas sensibles genera otros tres polígonos. El hospital de la parte superior izquierda forma intersección dos veces, mientras que el colegio de la parte inferior derecha forma intersección una sola vez.

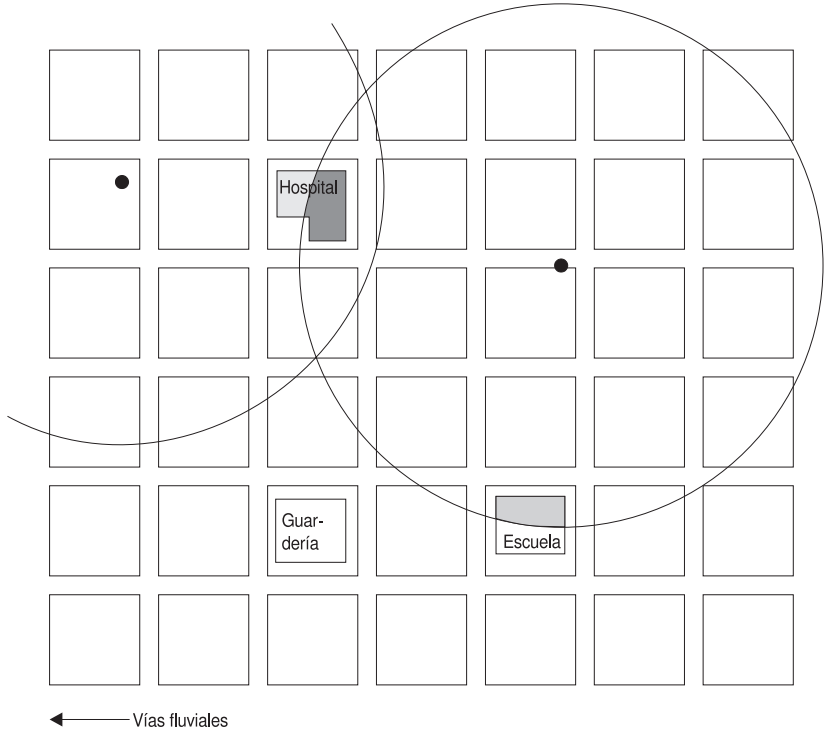


Figura 35. Utilización de ST_Intersection para determinar en qué medida un área de cada uno de los edificios puede verse afectada por los residuos peligrosos

ST_Intersects

ST_Intersects utiliza como entrada dos geometrías y devuelve un 1 (TRUE) si la intersección de las dos geometrías no da como resultado un conjunto vacío. De lo contrario, devuelve un 0 (FALSE).

Sintaxis

db2gse.ST_Intersects(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

El jefe de bomberos necesita una lista de todas las áreas sensibles que quedan dentro de un radio de cinco millas de un sitio de residuos peligrosos.

Las áreas sensibles se guardan en la tabla SENSITIVE_AREAS que se crea con la siguiente sentencia CREATE TABLE. La columna ZONE está definida como un polígono que guarda el contorno de cada una de las áreas sensibles.

```
CREATE TABLE SENSITIVE_AREAS (id       integer,
                              name     varchar(128),
                              size     float,
                              type     varchar(10),
                              zone     db2gse.ST_Polygon);
```

Los sitios peligrosos se guardan en la tabla HAZARDOUS_SITES, creada con la siguiente sentencia CREATE TABLE. La columna LOCATION, definida como un punto, almacena la ubicación que constituye el centro geográfico de cada sitio peligroso.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(128),
                              location  db2gse.ST_Point);
```

La consulta devuelve una lista de áreas sensibles y nombres de sitios peligrosos correspondientes a áreas sensibles que forman intersección con la zona de influencia de cinco millas que rodea los sitios peligrosos.

```
SELECT sa.name, hs.name
FROM SENSITIVE_AREAS sa, HAZARDOUS_SITES hs
WHERE db2gse.ST_Intersects(db2gse.ST_Buffer(hs.location,(5 * 5280)),sa.zone)
= 1;
```

ST_IsClosed

ST_IsClosed utiliza como entrada una línea o multilínea y devuelve un 1 (TRUE) si está cerrada y 0 (FALSE) si no es así.

Sintaxis

db2gse.ST_IsClosed(c db2gse.ST_Curve)
db2gse.ST_IsClosed(mc db2gse.ST_MultiCurve)

Tipo devuelto

Integer

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla CLOSED_LINESTRING, que contiene una sola columna de líneas.

```
CREATE TABLE CLOSED_LINESTRING (ln1 db2gse.ST_LineString)
```

Las siguientes sentencias INSERT insertan dos registros en la tabla CLOSED_LINESTRING. El primer registro no es una línea cerrada y el segundo sí lo es.

```
INSERT INTO CLOSED_LINESTRING
VALUES(db2gse.ST_LineFromText('linestring  (10.02 20.01,10.32 23.98,
       11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO CLOSED_LINESTRING
VALUES(db2gse.ST_LineFromText('linestring  (10.02 20.01,11.92 35.64,
       25.02 34.15, 19.15 33.94,10.02 20.01)',
                     db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante asociado muestran los resultados de la función ST_IsClosed. La primera fila devuelve un 0 porque la línea no está cerrada, mientras que la segunda fila devuelve un 1 porque la línea está cerrada.

```
SELECT db2gse.ST_IsClosed(ln1) "Is it closed" FROM CLOSED_LINESTRING

Is it closed
------------
           0
           1

  2 record(s) selected.
```

La siguiente sentencia CREATE TABLE crea la tabla CLOSED_MULTILINESTRING, que tiene una sola columna de multilínea.

```
CREATE TABLE CLOSED_MULTILINESTRING (mln1 db2gse.ST_MultiLineString)
```

Las siguientes sentencias INSERT insertan dos registros en CLOSED_MULTILINESTRING, un registro de multilínea que no está cerrado y otro que sí lo está.

```
INSERT INTO CLOSED_MULTILINESTRING
VALUES(db2gse.ST_MLineFromText('multilinestring ((10.02 20.01,10.32 23.98,
       11.92 25.64), (9.55 23.75,15.36 30.11))',
                      db2gse.coordref()..srid(0)))

INSERT INTO CLOSED_MULTILINESTRING
VALUES(db2gse.ST_MLineFromText('multilinestring ((10.02 20.01,11.92 35.64,
       25.02 34.15, 19.15 33.94,10.02 20.01),
                                       (51.71 21.73,73.36 27.04,71.52 32.87,
                                        52.43 31.90,51.71 21.73))',
                      db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante asociado muestran los resultados de la función ST_IsClosed. La primera fila devuelve un 0 porque la multilínea no está cerrada, mientras que la segunda fila devuelve un 1 porque la multilínea está cerrada. Una multilínea está cerrada si todas las líneas que la componen están cerradas.

```
SELECT db2gse.ST_IsClosed(mln1) "Is it closed" FROM CLOSED_MULTILINESTRING

Is it closed
------------
           0
           1

  2 record(s) selected.
```

ST_IsEmpty

ST_IsEmpty utiliza como entrada un objeto de geometría y devuelve un 1 (TRUE) si está vacío y 0 (FALSE) si no es así.

Sintaxis

db2gse.ST_IsEmpty(g db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla EMPTY_TEST con dos columnas. La columna GEOTYPE guarda el tipo de datos de las subclases que se guardan en la columna de geometría G1.

```
CREATE TABLE EMPTY_TEST (geotype varchar(20), g1 db2gse.ST_Geometry)
```

Las siguientes sentencias INSERT insertan dos registros para las subclases de geometría punto, línea y polígono. Un registro está vacío y el otro no.

```
INSERT INTO EMPTY_TEST
VALUES('Point', db2gse.ST_PointFromText('point (10.02 20.01)',
db2gse.coordref()..srid(0)))

INSERT INTO EMPTY_TEST
VALUES('Point', db2gse.ST_PointFromText('point empty',
db2gse.coordref()..srid(0)))

INSERT INTO EMPTY_TEST
VALUES('Linestring', db2gse.ST_LineFromText('linestring (10.02 20.01,
       10.32 23.98, 11.92 25.64)',
                                   db2gse.coordref()..srid(0)))

INSERT INTO EMPTY_TEST
VALUES('Linestring', db2gse.ST_LineFromText('linestring  empty',
db2gse.coordref()..srid(0)))

INSERT INTO EMPTY_TEST
VALUES('Polygon', db2gse.ST_PolyFromText('polygon ((10.02 20.01,11.92 35.64,
25.02 34.15,19.15 33.94,10.02 20.01))',
                               db2gse.coordref()..srid(0)))

INSERT INTO EMPTY_TEST
VALUES('Polygon', db2gse.ST_PolyFromText('polygon  empty',
db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente muestran el tipo de geometría de la columna GEOTYPE y los resultados de la función ST_IsEmpty.

```
SELECT geotype, db2gse.ST_IsEmpty(g1) "It is empty" FROM EMPTY_TEST

GEOTYPE                   It is empty
--------------------      -----------
ST_Point                    0
ST_Point                    1
ST_Linestring               0
ST_Linestring               1
ST_Polygon                  0
ST_Polygon                  1

  6 record(s) selected.
```

ST_IsRing

ST_IsRing utiliza como entrada una línea y devuelve un 1 (TRUE) si es un anillo (es decir, la línea está cerrada y es simple) y un 0 (FALSE) en caso contrario.

Sintaxis

db2gse.ST_IsRing(c db2gse.ST_Curve)

Tipo devuelto

Integer

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla RING_LINESTRING, que tiene una sola columna de líneas denominada LN1.

```
CREATE TABLE RING_LINESTRING (ln1 db2gse.ST_LineString)
```

Las siguientes sentencias INSERT insertan tres líneas en la columna LN1. La primera fila contiene una línea que no está cerrada y por lo tanto no es un anillo. La segunda fila contiene una línea que está cerrada y es simple, por lo tanto es un anillo. La tercera fila contiene una línea que está cerrada pero no es simple porque forma intersección con su propio interior, por lo tanto no es un anillo.

```
INSERT INTO RING_LINESTRING
VALUES(db2gse.ST_LineFromText('linestring  (10.02 20.01,10.32 23.98,
        11.92 25.64)', db2gse.coordref()..srid(0)))

INSERT INTO RING_LINESTRING
VALUES(db2gse.ST_LineFromText('linestring (10.02 20.01,11.92 35.64,25.02 34.15,
                                19.15 33.94, 10.02 20.01)',
                    db2gse.coordref()..srid(0)))

INSERT INTO RING_LINESTRING
VALUES(db2gse.ST_LineFromText('linestring (15.47 30.12,20.73 22.12,10.83 14.13,
                                16.45 17.24,21.56 13.37,11.23 22.56,
                                19.11 26.78,15.47 30.12)',
                    db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente muestran los resultados de la función ST_IsRing. Las filas primera y tercera devuelven un 0. Esto se debe a que las líneas no son anillos, mientras que la segunda fila devuelve un 1 porque es un anillo.

```
SELECT db2gse.ST_IsRing(ln1) "Is it ring" FROM RING_LINESTRING

Is it ring
-----------
          0
          1
          0

  3 record(s) selected.
```

ST_IsSimple

ST_IsSimple utiliza como entrada un objeto de geometría y devuelve un 1 (TRUE) si el objeto es simple y un 0 (FALSE) en caso contrario.

Sintaxis

db2gse.ST_IsSimple(g db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla ISSIMPLE_TEST, que tiene dos columnas. La columna PID, que es de tipo smallint, contiene el identificador exclusivo de cada fila. La columna de geometría G1 guarda los ejemplos de geometrías simples y no simples.

```
CREATE TABLE ISSIMPLE_TEST (pid smallint, g1 db2gse.ST_Geometry)
```

Las siguientes sentencias INSERT insertan dos registros en la tabla ISSIMPLE_TEST. El primero es simple porque es una línea que no forma intersección con su interior. El segundo no es simple porque forma intersección con su interior.

```
INSERT INTO ISSIMPLE_TEST
VALUES (1, db2gse.ST_LineFromText('linestring (10 10, 20 20, 30 30)',
db2gse.coordref()..srid(0)))

INSERT INTO ISSIMPLE_TEST
VALUES (2, db2gse.ST_LineFromText('linestring (10 10, 20 20,20 30,10 30,10 20,
20 10)', db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente muestran los resultados de la función ST_IsSimple. El primer registro devuelve un 1 porque la línea es simple, mientras que el segundo registro devuelve un 0 porque la línea no es simple.

```
SELECT ST_IsSimple(g1)
FROM ISSIMPLE_TEST

g1
--------------
1
0
```

ST_IsValid

ST_IsValid utiliza como entrada un valor ST_Geometry y devuelve un 1 (TRUE) si es válido y 0 (FALSE) si no lo es. Una geometría insertada en una base de datos DB2 siempre será válida porque Spatial Extender siempre valida sus datos espaciales antes de aceptarlos. Sin embargo, puede que otros proveedores de DBMS no validen la entrada, por lo que la aplicación debe hacerlo.

Sintaxis

db2gse.ST_IsValid(g db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La tabla valid_test se crea con las columnas geotype y g1. La columna geotype guarda el nombre de la subclase de geometría guardada en la columna de geometría g1.

```
CREATE TABLE valid_test (geotype varchar(20), g1 db2gse.ST_Geometry)
```

Las sentencias INSERT insertan una subclase de ejemplo en la tabla valid_test.

```
INSERT INTO valid_test VALUES(
    'Point', db2gse.ST_PointFromText('point (10.02 20.01)',
    db2gse.coordref()..srid(0))
)

INSERT INTO valid_test VALUES(
'Linestring',
db2gse.ST_LineFromText('linestring (10.02 20.01,10.32 23.98,11.92 25.64)',
db2gse.coordref()..srid(0))
)

INSERT INTO valid_test VALUES(
'Polygon', db2gse.ST_PolyFromText('polygon ((10.02 20.01,11.92 35.64,
    25.02 34.15, 19.15 33.94,10.02 20.01))', db2gse.coordref()..srid(0))
)

INSERT INTO valid_test VALUES(
'Multipoint', db2gse.ST_MPointFromText('multipoint (10.02 20.01,10.32 23.98,
11.92 25.64)', db2gse.coordref()..srid(0))
)

INSERT INTO valid_test VALUES(
'Multilinestring', db2gse.ST_MLineFromText('multilinestring ((10.02 20.01,
10.32 23.98,11.92 25.64),(9.55 23.75,15.36 30.11))',
db2gse.coordref()..srid(0))
)

INSERT INTO valid_test VALUES(
'Multipolygon',
```

```
db2gse.ST_MPolyFromText('multipolygon (((10.02 20.01,11.92 35.64,
25.02 34.15,19.15 33.94,10.02 20.01)),((51.71 21.73,73.36 27.04,71.52 32.87,
52.43 31.90,51.71 21.73)))', db2gse.coordref()..srid(0))
)
```

La sentencia SELECT lista el nombre de subclase almacenado en la columna GEOTYPE (tipo de geometría) junto con la dimensión de ese tipo de geometría.

```
SELECT geotype, db2gse.ST_IsValid(g1) Valid FROM valid_test

GEOTYPE              Valid
-------------------- -------------
ST_Point                       1
ST_Linestring                  1
ST_Polygon                     1
ST_Multipoint                  1
ST_Multilinestring             1
ST_Multipolygon                1

6 record(s) selected.
```

ST_Length

ST_Length utiliza como entrada una línea o multilínea y devuelve su longitud.

Sintaxis

db2gse.ST_Length(c db2gse.ST_Curve)
db2gse.ST_Length(mc db2gse.ST_MultiCurve)

Tipo devuelto

Double

Ejemplos

Un ecologista local está estudiando los patrones de migración de la población de salmones de las vías fluviales de la región. El ecologista desea saber la longitud de todos los sistemas de ríos y corrientes que fluyen por la región.

La siguiente sentencia CREATE TABLE crea la tabla WATERWAYS. Las columnas ID y NAME identifican cada sistema de corrientes y de ríos que se guarda en la tabla. La columna WATER es una multilínea, pues los sistemas de ríos y de corrientes suelen ser un agregado de varias líneas.

```
CREATE TABLE WATERWAYS (id integer, name varchar(128),
water db2gse.ST_MultiLineString);
```

La siguiente sentencia SELECT utiliza la función ST_Length para devolver el nombre y la longitud de cada vía fluvial de la región.

```
SELECT name, db2gse.ST_Length(water) "Length"
FROM WATERWAYS;
```

La Figura 36 en la página 279 muestra los sistemas de ríos y corrientes que fluyen dentro de los límites de la región.

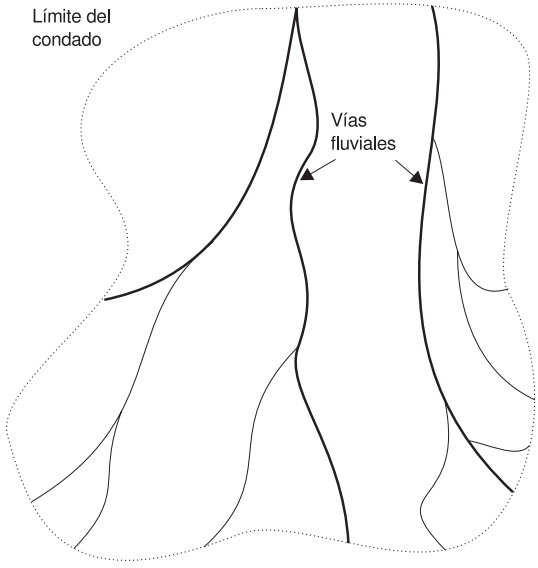


Figura 36. Utilización de ST_Length para determinar la longitud total de las vías fluviales de una región

ST_LineFromText

ST_LineFromText utiliza como entrada una representación de texto convencional del tipo línea y un identificador de sistema de referencias espaciales, y devuelve una línea.

Sintaxis

db2gse.ST_LineFromText(lineStringTaggedText Varchar(4000), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_LineString

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla LINESTRING_TEST, que contiene una sola columna de líneas LN1.

```
CREATE TABLE LINESTRING_TEST (ln1 db2gse.ST_LineString)
```

La siguiente sentencia INSERT inserta una línea en la columna LN1 utilizando la función ST_LineFromText.

```
INSERT INTO LINESTRING_TEST
VALUES (db2gse.ST_LineFromText('linestring(10.01 20.03,20.94 21.34,
        35.93 19.04)', db2gse.coordref()..srid(0)))
```

ST_LineFromWKB

ST_LineFromWKB utiliza como entrada una representación binaria convencional del tipo línea y un identificador de sistema de referencias espaciales, y devuelve una línea.

Sintaxis

db2gse.ST_LineFromWKB(WKBLineString Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_LineString

Ejemplos

El siguiente fragmento de código llena la tabla SEWERLINES con el id exclusivo, clase de tamaño y geometría de cada línea de alcantarillado.

La tabla SEWERLINES se crea con tres columnas. La primera columna, SEWER_ID, identifica de forma exclusiva cada línea de alcantarillado. La segunda columna, CLASS, de tipo entero, identifica el tipo de línea de alcantarillado, que suele estar asociado a la capacidad de la línea. La tercera columna, SEWER, de tipo línea, guarda la geometría de la línea de alcantarillado.

```
CREATE TABLE SEWERLINES (sewer_id   integer,
                         class  integer,
                         sewer  db2gse.ST_LineString);

/* Crear la sentencia SQL insert para llenar el id de alcantarillado,
   la clase de tamaño y la línea del alcantarillado.
   Los interrogantes son marcadores de parámetro que indican los valores
   de id de alcantarillado, clase y geometría de alcantarillado que se
   recuperarán en el momento de la ejecución. */
strcpy (wkb_sql,"insert into sewerlines (sewer_id,class,sewer)
values (?,?, db2gse.ST_LineFromWKB (cast(? as blob(1m)),
        db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)wkb_sql, SQL_NTS);

/* Vincular el valor entero sewer_id con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_SLONG,
   SQL_INTEGER, 0, 0, &sewer_id, 0, &pcbvalue1);

/* Vincular el valor entero class con el segundo parámetro. */
pcbvalue2 = 0;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_SLONG,
   SQL_INTEGER, 0, 0, &sewer_class, 0, &pcbvalue2);
```

```
/* Vincular la forma con el tercer parámetro. */
pcbvalue3 = blob_len;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, blob_len, 0, sewer_wkb, blob_len, &pcbvalue3);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

ST_MLineFromText

ST_MLineFromText utiliza como entrada una representación de texto convencional del tipo multilínea y un identificador de sistema de referencias espaciales, y devuelve una multilínea.

Sintaxis

db2gse.ST_MLineFromText(multiLineStringTaggedText String, SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiLineString

Ejemplos

la siguiente sentencia CREATE TABLE crea la tabla MLINESTRING_TEST. MLINESTRING_TEST tiene dos columnas: la columna tipo smallint GID, que identifica de forma exclusiva la fila, y la columna de multilínea ML1.

```
CREATE TABLE ST_MLINESTRING_TEST (gid smallint, ml1 db2gse.ST_MultiLineString)
```

La siguiente sentencia INSERT inserta la multilínea con la función ST_MLineFromText.

```
INSERT INTO MLINESTRING_TEST
VALUES (1, db2gse.ST_MLineFromText('multilinestring((10.01 20.03,10.52 40.11,
         30.29 41.56,31.78 10.74),
                                        (20.93 20.81, 21.52 40.10))',
                          db2gse.coordref()..srid(0)))
```

ST_MLineFromWKB

ST_MLineFromWKB utiliza como entrada una representación binaria convencional del tipo multilínea y un identificador de sistema de referencias espaciales, y devuelve una multilínea.

Sintaxis

db2gse.ST_MLineFromWKB(WKBMultiLineString Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiLineString

Ejemplos

El siguiente fragmento de código llena la tabla WATERWAYS con un ID exclusivo, un nombre y una multilínea denominada WATER.

La tabla WATERWAYS se crea con las columnas ID y NAME, que identifican cada sistema de corrientes y de ríos guardado en la tabla. La columna WATER es una multilínea, pues los sistemas de ríos y de corrientes suelen ser un agregado de varias líneas.

```
CREATE TABLE WATERWAYS (id        integer,
                        name      varchar(128),
                        water     db2gse.ST_MultiLineString);

/* Crear la sentencia INSERT de SQL para entrar datos en el
ID, el nombre y la multilínea. Los interrogantes son marcadores de
parámetros que indican el ID, el nombre y los valores de WATER que se
recuperarán durante la ejecución. */
strcpy (shp_sql,"insert into WATERWAYS (id,name,water)
values (?,?, db2gse.ST_MLineFromWKB (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)shp_sql, SQL_NTS);

/* Vincular el valor entero id con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_SLONG,
   SQL_INTEGER, 0, 0, &id, 0, &pcbvalue1);

/* Vincular el valor varchar name con el segundo parámetro. */
pcbvalue2 = name_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_CHAR,
     SQL_CHAR, name_len, 0, &name, name_len, &pcbvalue2);

/* Vincular la forma con el tercer parámetro. */
pcbvalue3 = blob_len;
```

```
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, blob_len, 0, water_shape, blob_len, &pcbvalue3);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

ST_MPointFromText

ST_MPointFromText utiliza como entrada una representación de texto convencional del tipo multipunto y un identificador de sistema de referencias espaciales, y devuelve un multipunto.

Sintaxis

db2gse.ST_MPointFromText(multiPointTaggedText Varchar(4000), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiPoint

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla MULTIPOINT_TEST con una sola columna multipunto, MPT1.

```
CREATE TABLE MULTIPOINT_TEST (mpt1 db2gse.ST_MultiPoint)
```

La siguiente sentencia INSERT inserta un multipunto en la columna MPT1 utilizando la columna ST_MPointFromText.

```
INSERT INTO MULTIPOINT_TEST
VALUES (1, db2gse.ST_MPointFromText('multipoint(10.01 20.03,10.52 40.11,
                                    30.29 41.56,31.78 10.74)',
                         db2gse.coordref()..srid(0)))
```

ST_MPointFromWKB

ST_MPointFromWKB utiliza como entrada una representación binaria convencional del tipo multipunto y un identificador de sistema de referencias espaciales, y devuelve un multipunto.

Sintaxis

db2gse.ST_MPointFromWKB(WKBMultiPoint Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiPoint

Ejemplos

El siguiente fragmento de código llena la tabla SPECIES_SITINGS.

La tabla SPECIES_SITINGS se crea con tres columnas. Las columnas SPECIES y GENUS identifican de forma exclusiva cada fila, mientras que la columna multipunto SITINGS guarda las ubicaciones de las especies.

```
CREATE TABLE SPECIES_SITINGS (species  varchar(32),
                              genus varchar(32),
                              sitings  db2gse.ST_MultiPoint);

/* Crear la sentencia insert de SQL para entrar datos en
SPECIES, GENUS y SITINGS. Los interrogantes son marcadores de parámetro
que indican los valores de SPECIES, GENUS y SITINGS que se recuperarán
durante la ejecución. */
strcpy (wkb_sql,"insert into SPECIES_SITINGS (species,genus,sitings)
values (?,?, db2gse.ST_MPointFromWKB (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y asociar
el descriptor de sentencia con el descriptor de conexión. */
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)wkb_sql, SQL_NTS);

/* Vincular el valor varchar species con el primer parámetro. */
pcbvalue1 = species_len;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_CHAR,
   SQL_CHAR, species_len, 0, &species, species_len, &pcbvalue1);
/* Vincular el valor varchar genus con el segundo parámetro. */
pcbvalue2 = genus_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_CHAR,
   SQL_CHAR, genus_len, 0, &name, genus_len, &pcbvalue2);

/* Vincular la forma con el tercer parámetro. */
pcbvalue3 = sitings_len;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, sitings_len, 0, sitings_wkb, sitings_len, &pcbvalue3);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

ST_MPolyFromText

ST_MPolyFromText utiliza como entrada una representación de texto convencional del tipo multipolígono y un identificador de sistema de referencias espaciales, y devuelve un multipolígono.

Esta función no puede utilizar como entrada un multipolígono que contenga varios polígonos que tienen las mismas coordenadas.

Sintaxis

db2gse.ST_MPolyFromText(multiPolygonTaggedText Varchar(4000), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiPolygon

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla MULTIPOLYGON_TEST que tiene una sola columna de multipolígono, MPL1.

```
CREATE TABLE MULTIPOLYGON_TEST (mpl1 db2gse.ST_MultiPolygon)
```

La siguiente sentencia INSERT inserta un multipolígono en la columna MPL1 utilizando la función ST_MPolyFromText.

```
INSERT INTO MULTIPOLYGON_TEST VALUES (
db2gse.ST_MPolyFromText('multipolygon(((10.01 20.03,10.52 40.11,
 30.29 41.56,31.78 10.74,10.01 20.03),(21.23 15.74,21.34 35.21,28.94 35.35,
 29.02 16.83, 21.23 15.74)),((40.91 10.92,40.56 20.19,
 50.01 21.12,51.34 9.81, 40.91 10.92)))',

db2gse.coordref()..srid(0)))
```

ST_MPolyFromWKB

ST_MPolyFromWKB utiliza como entrada una representación binaria convencional del tipo multipolígono y un identificador de sistema de referencias espaciales, y devuelve un multipolígono.

Sintaxis

db2gse.ST_MPolyFromWKB(WKBMultiPolygon Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_MultiPolygon

Ejemplos

El siguiente fragmento de código llena la tabla LOTS.

La tabla LOTS guarda la columna LOT_ID, que identifica de forma exclusiva cada parcela, y el multipolígono LOT, que contiene la geometría de línea de la parcela.

```
CREATE TABLE LOTS (  lot_id integer, lot db2gse.ST_MultiPolygon );

/* Crear la sentencia insert de SQL para entrar datos en el
ID de parcela y el multipolígono de parcela. Los interrogantes son
marcadores de parámetro que indican el ID de parcela y los valores de
parcela que se recuperarán durante la ejecución. */
strcpy (wkb_sql,"insert into LOTS (lot_id,lot)
values (?, db2gse.ST_MPolyFromWKB (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)wkb_sql, SQL_NTS);

/* Vincular el valor entero lot_id con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_INTEGER,
     SQL_INTEGER, 0, 0, &lot_id, 0, &pcbvalue1);

/* Vincular la forma de parcela con el segundo parámetro. */
pcbvalue2 = lot_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, lot_len, 0, lot_wkb, lot_len, &pcbvalue2);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

ST_NumGeometries

ST_NumGeometries utiliza como entrada un grupo y devuelve el número de geometrías del grupo.

Sintaxis

db2gse.ST_NumGeometries(g db2gse.ST_GeomCollection)

Tipo devuelto

Integer

Ejemplos

El ingeniero municipal desea saber el número real de edificios de cada zona edificada.

Las zonas edificadas se guardan en la tabla BUILDINGFOOTPRINTS que se creó con la siguiente sentencia CREATE TABLE.

```
CREATE TABLE BUILDINGFOOTPRINTS (   building_id integer,
                                    lot_id      integer,
                                    footprint   db2gse.ST_MultiPolygon);
```

La siguiente sentencia SELECT utiliza la función ST_NumGeometries para listar el BUILDING_ID que identifica de forma exclusiva cada edificio y el número de edificios de cada zona edificada.

```
SELECT building_id, db2gse.ST_NumGeometries (footprint) "Number of buildings"
FROM BUILDINGFOOTPRINTS;
```

ST_NumInteriorRing

ST_NumInteriorRing utiliza como entrada un polígono y devuelve el número de anillos interiores que contiene.

Sintaxis

db2gse.ST_NumInteriorRing(p db2gse.ST_Polygon)

Tipo devuelto

Integer

Ejemplos

Un ornitólogo que está estudiando la población de aves en varias islas de los mares del sur sabe que la zona en la que se alimenta una determinada especie se limita a las islas que contienen lagos de agua fresca. Por lo tanto, desea saber qué islas contienen uno o más lagos.

La siguiente sentencia CREATE TABLE crea la tabla ISLANDS. Las columnas ID y NAME de la tabla ISLANDS identifican cada isla y la columna tipo polígono LAND guarda la geometría de la isla.

```
CREATE TABLE ISLANDS (id integer, name varchar(32), land db2gse.ST_Polygon);
```

Puesto que los anillos interiores representan lagos, se utiliza la función ST_NumInteriorRing para listar únicamente las islas que tienen al menos un anillo interior.

```
SELECT name FROM ISLANDS WHERE db2gse.ST_NumInteriorRing(land) > 0;
```

ST_NumPoints

ST_NumPoints utiliza como entrada una línea y devuelve el número de puntos que contiene.

Sintaxis

db2gse.ST_NumPoints(l db2gse.ST_LineString)

Tipo devuelto

Integer

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla NUMPOINTS_TEST. La columna GEOTYPE contiene el tipo de geometría guardado en la columna de geometría G1.

```
CREATE TABLE NUMPOINTS_TEST (geotype varchar(12), g1 db2gse.ST_Geometry)
```

La siguiente sentencia INSERT inserta una línea.

```
INSERT INTO NUMPOINTS_TEST VALUES( linestring,
db2gse.ST_LineFromText('linestring (10.02 20.01, 23.73 21.92)',
db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente listan el tipo de geometría y el número de puntos que contiene cada una.

```
SELECT geotype, db2gse.ST_NumPoints(g1)
FROM NUMPOINTS_TEST

GEOTYPE      Number of points
------------ ----------------
ST_linestring          2
  1 record(s) selected.
```

ST_OrderingEquals

ST_OrderingEquals compara dos geometrías y devuelve un 1 (TRUE) si las geometrías son iguales y las coordenadas están en el mismo orden y 0 (FALSE) si no es así.

Sintaxis

db2gse.ST_OrderingEquals(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla LINESTRING_TEST, que tiene dos columnas tipo línea, L1 y L2.

```
CREATE TABLE LINESTRING_TEST (lid integer, l1 db2gse.ST_LineString,
l2 db2gse.ST_LineString);
```

La siguiente sentencia INSERT inserta en L1 y L2 dos líneas que son iguales y tienen el mismo orden de coordenadas.

```
INSERT INTO linestring_test VALUES (1,
db2gse.LineFromText('linestring (10.01 20.02, 21.50 12.10)',
  db2gse.coordref()..srid(0)),
db2gse.LineFromText('linestring (10.01 20.02, 21.50 12.10)',
  db2gse.coordref()..srid(0)));
```

La siguiente sentencia INSERT inserta en L1 y L2 dos líneas que son iguales pero no tienen el mismo orden de coordenadas.

```
INSERT INTO linestring_test VALUES (2,
db2gse.LineFromText('linestring (10.01 20.02, 21.50 12.10)',
  db2gse.coordref()..srid(0)),
db2gse.LineFromText('linestring (21.50 12.10,10.01 20.02)',
  db2gse.coordref()..srid(0)));
```

Tal como muestran la sentencia SELECT y el conjunto resultante mostrados a continuación, la función ST_OrderingEquals:

- Devuelve un 1 (TRUE) cuando las geometrías de entrada son iguales y sus coordenadas están en el mismo orden
- Devuelve un 0 (FALSE) cuando las geometrías de entrada no son iguales o cuando las coordenadas de las geometrías están en un orden diferente la una respecto de la otra

```
SELECT lid, db2gse.ST_OrderingEquals(l1,l2) OrderingEquals FROM linestring_test

lid   OrderingEquals
---    -----------
 1           1
 2           0
```

ST_Overlaps

ST_Overlaps utiliza como entrada dos objetos de geometría y devuelve un 1 (TRUE) si la intersección entre los objetos da como resultado un objeto de geometría de la misma dimensión pero que no es igual a ninguno de los dos objetos fuente; si no es así, devuelve un 0 (FALSE).

Sintaxis

db2gse.ST_Overlaps(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

El supervisor de un condado necesita una lista de sitios de residuos peligrosos cuyo radio de cinco millas se solape con áreas sensibles.

La siguiente sentencia CREATE TABLE crea la tabla SENSITIVE_AREAS. La tabla SENSITIVE_AREAS contiene varias columnas que describen las instituciones amenazadas además de la columna ZONE, que guarda la geometría de polígono de la institución.

```
CREATE TABLE SENSITIVE_AREAS (id        integer,
                              name      varchar(128),
                              size      float,
                              type      varchar(10),
                              zone      db2gse.ST_Polygon);
```

La tabla HAZARDOUS_SITES contiene el identificador de los sitios en las columnas SITE_ID y NAME, mientras que la ubicación geográfica real de cada sitio se guarda en la columna LOCATION, de tipo punto.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(128),
                              location  db2gse.ST_Point);
```

En la siguiente sentencia SELECT, las tablas SENSITIVE_AREAS y HAZARDOUS_SITES se unen mediante la función ST_Overlaps. Devuelve un 1 (TRUE) para todas las filas de la tabla SENSITIVE_AREAS cuyos polígonos de zona se solapan con el radio de cinco millas que rodea el punto de cada ubicación HAZARDOUS_SITES.

```
SELECT hs.name
FROM HAZARDOUS_SITES hs, SENSITIVE_AREAS sa
WHERE db2gse.ST_Overlaps (buffer(hs.location,(5 * 5280)),sa.zone) = 1;
```

En la Figura 37 en la página 295, el hospital y el colegio se solapan con el radio de cinco millas de dos zonas de residuos peligrosos del condado, mientras que la guardería no se solapa con ninguno.

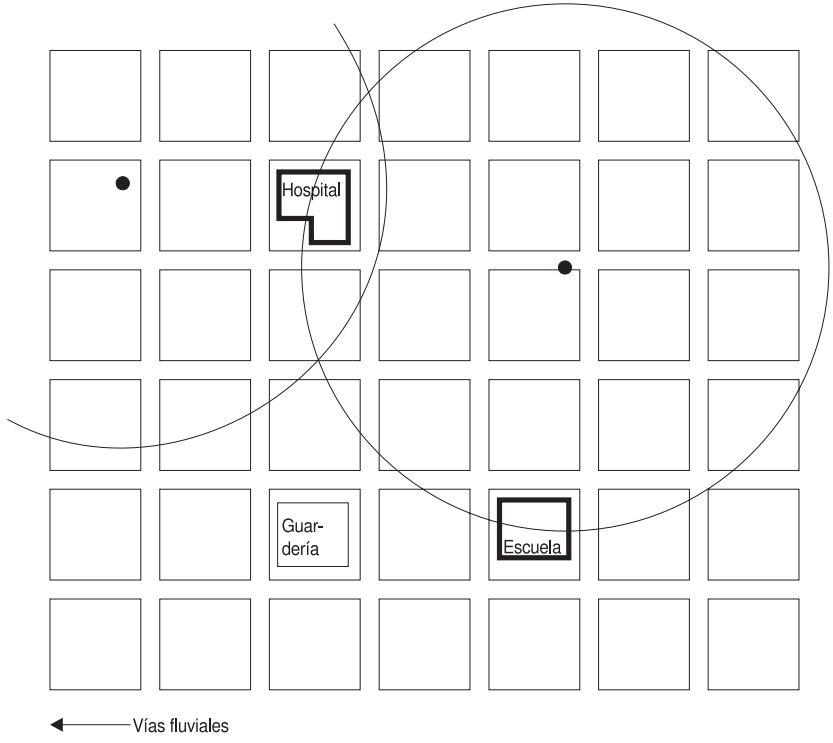


Figura 37. Utilización de ST_Overlaps para determinar los edificios que se encuentran, al menos parcialmente, dentro de un área de residuos peligrosos

ST_Perimeter

ST_Perimeter devuelve el perímetro de un polígono.

Sintaxis

db2gse.ST_Perimeter(p db2gse.ST_Polygon)

Tipo devuelto

Double

Ejemplos

Un ecologista que estudia las aves que habitan en las orillas de masas de agua tiene que determinar el perímetro de los lagos de una determinada zona. Los lagos se almacenan como polígonos en la tabla WATERBODIES, que se creó con la siguiente sentencia CREATE TABLE.

```
CREATE TABLE WATERBODIES (wbid  integer,
                          waterbody  db2gse.ST_MultiPolygon);
```

En la siguiente sentencia SELECT, la función ST_Perimeter devuelve el perímetro que rodea cada zona con agua, mientras que la función SUM suma los perímetros para devolver el total.

```
SELECT SUM(db2gse.ST_Perimeter(waterbody))
FROM waterbodies;
```

ST_Point

ST_Point devuelve un ST_Point a partir de una coordenada y, una coordenada x y una referencia espacial determinadas.

Sintaxis

db2gse.ST_Point(X Double, Y Double, SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Point

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla POINT_TEST, que contiene una sola columna de tipo de punto, PT1.

```
CREATE TABLE POINT_TEST (pt1 db2gse.ST_Point)
```

La función ST_Point convierte las coordenadas de punto en una geometría punto antes de que la sentencia INSERT lo inserte en la columna PT1.

```
INSERT INTO point_test VALUES(
    db2gse.ST_Point(10.01,20.03, db2gse.coordref()..srid(0))
)
```

ST_PointFromText

ST_PointFromText utiliza como entrada una representación de texto convencional del tipo punto y un identificador de sistema de referencias espaciales, y devuelve un punto.

Sintaxis

db2gse.ST_PointFromText(pointTaggedText Varchar(4000), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Point

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla POINT_TEST, que contiene una sola columna de tipo punto, PT1.

```
CREATE TABLE POINT_TEST (pt1 db2gse.ST_Point)
```

Antes de que la sentencia INSERT inserte el punto en la columna PT1, la función ST_PointFromText convierte las coordenadas del texto del punto en el formato del punto.

```
INSERT INTO POINT_TEST VALUES (
       db2gse.ST_PointFromText ('point(10.01 20.03)',
         db2gse.coordref()..srid(0)))
```

ST_PointFromWKB

ST_PointFromWKB utiliza como entrada una representación binaria convencional del tipo punto y un identificador de sistema de referencias espaciales, y devuelve un punto.

Sintaxis

db2gse.ST_PointFromWKB(WKBPoint Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Point

Ejemplos

El siguiente fragmento de código llena la tabla HAZARDOUS_SITES.

Los sitios peligrosos se guardan en la tabla HAZARDOUS_SITES, creada con la siguiente sentencia CREATE TABLE. La columna LOCATION, definida como un punto, almacena la ubicación que constituye el centro geográfico de cada sitio peligroso.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(128),
                              location  db2gse.ST_Point);

/* Crear la sentencia SQL insert para llenar las columnas site_id, name y
   location. Los interrogantes son marcadores de parámetro que indican los
   valores de site_id, name y location que se recuperarán en el momento de
   la ejecución. */
strcpy (wkb_sql,"insert into HAZARDOUS_SITES (site_id, name, location)
values (?,?, db2gse.ST_PointFromWKB(cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y asociar
el descriptor de sentencia con el descriptor de conexión. */
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)wkb_sql, SQL_NTS);

/* Vincular el valor entero site_id con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_INTEGER,
     SQL_INTEGER, 0, 0, &site_id, 0, &pcbvalue1);

/* Vincular el valor varchar name con el segundo parámetro. */
pcbvalue2 = name_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_CHAR,
   SQL_CHAR, 0, 0, name, 0, &pcbvalue2);

/* Vincular la forma location con el tercer parámetro. */
pcbvalue3 = location_len;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
```

```
        SQL_BLOB, location_len, 0, location_wkb, location_len, &pcbvalue3);

    /* Ejecutar la sentencia insert. */
    rc = SQLExecute (hstmt);
```

ST_PointN

ST_PointN utiliza como entrada una línea y un índice entero y devuelve un punto que es el vértice número n de la ruta de la línea.

Sintaxis

db2gse.ST_PointN(l db2gse.ST_Curve, n Integer)

Tipo devuelto

db2gse.ST_Point

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla POINTN_TEST, que tiene dos columnas: la columna GID, que identifica de forma exclusiva cada fila, y la columna tipo línea LN1.

```
CREATE TABLE POINTN_TEST (gid integer, ln1 db2gse.ST_LineString)
```

Las siguientes sentencias INSERT insertan dos valores de línea. La primera línea no tiene coordenadas Z ni medidas, mientras que la segunda línea tiene ambas cosas.

```
INSERT INTO POINTN_TEST VALUES(1,
db2gse.ST_LineFromText('linestring (10.02 20.01,23.73 21.92,30.10 40.23)',
db2gse.coordref()..srid(0)))

INSERT INTO POINTN_TEST VALUES(2,
db2gse.ST_LineFromText('linestring  zm (10.02 20.01 5.0 7.0,
     23.73 21.92 6.5 7.1,30.10 40.23 6.9 7.2)', db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente listan la columna GID y el segundo vértice de cada línea. La primera fila da como resultado un punto sin coordenada Z ni medida, mientras que la segunda fila da como resultado un punto con una coordenada Z y una medida. La función ST_PointN devuelve puntos con una coordenada Z o una medida si las hay en la línea fuente.

```
SELECT gid, CAST(db2gse.ST_AsText(db2gse.ST_PointN(ln1,2)) AS varchar(60))
"The 2nd vertice"
FROM POINTN_TEST

GID         The 2nd vertice
----------- ------------------------------------------------------------
          1 POINT ( 23.73000000 21.92000000)
          2 POINT ZM ( 23.73000000 21.92000000 7.00000000 7.10000000)

  2 record(s) selected.
```

ST_PointOnSurface

ST_PointOnSurface utiliza como entrada un polígono o multipolígono y devuelve un punto.

Sintaxis

db2gse.ST_PointOnSurface(s db2gse.ST_Surface)
db2gse.ST_PointOnSurface(ms db2gse.ST_MultiSurface)

Tipo devuelto

db2gse.ST_Point

Ejemplos

El ingeniero municipal necesita crear un punto identificador para cada una de las zonas edificadas.

Las zonas edificadas se guardan en la tabla BUILDINGFOOTPRINTS que se creó con la siguiente sentencia CREATE TABLE.

```
CREATE TABLE BUILDINGFOOTPRINTS (   building_id integer,
                                    lot_id      integer,
                                    footprint   db2gse.ST_MultiPolygon);
```

La función ST_PointOnSurface genera un punto que con toda seguridad está dentro de las zonas edificadas.

```
SELECT db2gse.ST_PointOnSurface(footprint) FROM BUILDINGFOOTPRINTS;
```

ST_PolyFromText

ST_PolyFromText utiliza como entrada una representación de texto convencional del tipo polígono y un identificador de sistema de referencias espaciales, y devuelve un polígono.

Sintaxis

db2gse.ST_PolyFromText(polygonTaggedText Varchar(4000), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Polygon

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla POLYGON_TEST con una sola columna de polígonos.

```
CREATE TABLE POLYGON_TEST (pl1 db2gse.ST_Polygon)
```

La siguiente sentencia INSERT inserta un polígono en la columna de polígonos mediante la función ST_PolyFromText.

```
INSERT INTO POLYGON_TEST VALUES (db2gse.ST_PolyFromText(
'polygon((10.01 20.03,10.52 40.11,30.29
41.56, 31.78 10.74,10.01 20.03))',
    db2gse.coordref()..srid(0)))
```

ST_PolyFromWKB

ST_PolyFromWKB utiliza como entrada una representación binaria convencional del tipo polígono y un identificador de sistema de referencias espaciales, y devuelve un polígono.

Sintaxis

db2gse.ST_PolyFromWKB(WKBPolygon Blob(1M), SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Polygon

Ejemplos

El siguiente fragmento de código llena la tabla SENSITIVE_AREAS.

La tabla SENSITIVE_AREAS contiene varias columnas que describen las instituciones amenazadas además de la columna zone, que guarda la geometría de polígono de la institución.

```
CREATE TABLE SENSITIVE_AREAS (id        integer,
                              name      varchar(128),
                              size      float,
                              type      varchar(10),
                              zone      db2gse.ST_Polygon);


/* Crear la sentencia SQL insert para llenar las columnas
   id, name, size, type y zone. Los interrogantes son marcadores de parámetro
   que indican los valores id, name, size, type y zone que se recuperarán
   en el momento de la ejecución. */
strcpy (shp_wkb,"insert into SENSITIVE_AREAS (id, name, size, type, zone)
values (?,?,?,?, db2gse.ST_PolyFromWKB (cast(? as blob(1m)),
db2gse.coordref()..srid(0)))");

/* Asignar memoria para el descriptor de sentencia SQL y asociar
el descriptor de sentencia con el descriptor de conexión. */
rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */
rc = SQLPrepare (hstmt, (unsigned char *)wkb_sql, SQL_NTS);

/* Vincular el valor entero id con el primer parámetro. */
pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_INTEGER,
   SQL_INTEGER, 0, 0, &id, 0, &pcbvalue1);

/* Vincular el valor varchar name con el segundo parámetro. */
pcbvalue2 = name_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_CHAR,
   SQL_CHAR, 0, 0, name, 0, &pcbvalue2);

/* Vincular el valor float size con el tercer parámetro. */
pcbvalue3 = 0;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_FLOAT,
```

```
     SQL_REAL, 0, 0, &size, 0, &pcbvalue3);

/* Vincular el valor varchar type con el cuarto parámetro. */
pcbvalue4 = type_len;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_CHAR,
     SQL_VARCHAR, type_len, 0, type, type_len, &pcbvalue4);

/* Vincular el polígono zone con el quinto parámetro. */
pcbvalue5 = zone_len;
rc = SQLBindParameter (hstmt, 3, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, zone_len, 0, zone_wkb, zone_len, &pcbvalue5);

/* Ejecutar la sentencia insert. */
rc = SQLExecute (hstmt);
```

ST_Polygon

ST_Polygon genera un ST_Polygon a partir de ST_LineString.

Sintaxis

db2gse.ST_Polygon(l db2gse.ST_LineString)

Tipo devuelto

db2gse.ST_Polygon

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla POLYGON_TEST, que tiene una sola columna, P1.

```
CREATE TABLE POLYGON_TEST (p1 db2gse.ST_polygon)
```

La siguiente sentencia INSERT convierte un anillo (una línea cerrada y simple) en un polígono y lo inserta en la columna P1 mediante la función ST_LineFromText dentro de la función ST_Polygon.

```
INSERT INTO POLYGON_TEST VALUES (
db2gse.ST_Polygon(db2gse.ST_LineFromText('linestring(10.01 20.03,20.94
21.34,35.93 10.04,10.01 20.03)', db2gse.coordref()..srid(0)))
)
```

ST_Relate

ST_Relate compara dos geometrías y devuelve un 1 (TRUE) si las geometrías cumplen con las condiciones especificadas por la serie de la matriz patrón DE-9IM; si no es así, devuelve un 0 (FALSE). Para obtener información sobre las matrices patrón DE-9IM, consulte la sección "Funciones de predicado" en la página 175.

Sintaxis

db2gse.ST_Relate(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry, patternMatrix CHAR(9))

Tipo devuelto

Integer

Ejemplos

Una matriz patrón DE-9IM es un dispositivo que sirve para comparar geometrías. Hay varios tipos de matrices. Por ejemplo, la matriz patrón de *igualdad* le indicará si hay dos geometrías iguales.

En este ejemplo, una matriz patrón de igualdad, que se muestra en la Tabla 57, se lee de izquierda a derecha y de arriba a abajo como una serie ("T*F**FFF* ").

Tabla 57. Matriz de patrón de igualdad

		b		
		Interior	**Límite**	**Exterior**
	Interior	T	*	F
a	**Límite**	*	*	F
	Exterior	F	F	*

Luego se crea la tabla RELATE_TEST con la siguiente sentencia CREATE TABLE.

```
CREATE TABLE RELATE_TEST (rid integer, g1 db2gse.ST_Geometry,
g2 db2gse.ST_Geometry, g3 db2gse.ST_Geometry);
```

Las siguientes sentencias INSERT insertan una subclase de ejemplo en la tabla RELATE_TEST.

```
INSERT INTO RELATE_TEST VALUES(
       1,
       db2gse.ST_PointFromText('point (10.02 20.01)',
            db2gse.coordref()..srid(0),
       db2gse.ST_PointFromText('point (10.02 20.01)',
            db2gse.coordref()..srid(0),
        db2gse.ST_PointFromText('point (30.01 20.01)',
            db2gse.coordref()..srid(0)
)
```

La siguiente sentencia SELECT y el correspondiente conjunto resultante listan el nombre de subclase guardado en la columna GEOTYPE (tipo de geometría) junto con la dimensión de ese tipo de geometría.

```
SELECT rid, relate(g1,g2, 'T*F**FFF*') equals FROM relate_test

RID      equals
-------- -----------
1        1

  1 record(s) selected.
```

ST_SRID

ST_SRID utiliza como entrada un objeto de geometría y devuelve su identificador de sistema de referencias espaciales.

Sintaxis

db2gse.ST_SRID(g1 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

La función ST_SRID devuelve el identificador del sistema de referencias espaciales asociado al valor ST_Geometry.

Por ejemplo, se utiliza un tipo geometría en una sentencia CREATE TABLE:

```
CREATE TABLE SRID_TEST(g1 db2gse.ST_Geometry)
```

En la siguiente sentencia INSERT, se inserta una geometría de punto ubicada en la coordenada 10.01,50.76 en la columna de geometría G1. Cuando se creó la geometría de punto mediante la función ST_PointFromText, se le asignó el valor 1 de srid.

```
INSERT INTO SRID_TEST
VALUES (db2gse.ST_PointFromText('point(10.01 50.76)',
                                db2gse.coordref()..srid(0)))
```

La función ST_SRID devuelve el identificador del sistema de referencias espaciales de la geometría especificada como entrada, tal como muestra la siguiente sentencia SELECT y el conjunto resultante correspondiente.

```
SELECT db2gse.ST_SRID(g1) FROM SRID_TEST

g1
--------------
0
```

ST_StartPoint

ST_StartPoint utiliza como entrada una línea y devuelve un punto que es el primer punto de la línea.

Sintaxis

db2gse.ST_StartPoint(c db2gse.ST_Curve)

Tipo devuelto

db2gse.ST_Point

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla STARTPOINT_TEST. STARTPOINT_TEST tiene dos columnas: la columna de enteros GID, que identifica de forma exclusiva las filas de la tabla, y la columna tipo línea LN1.

```
CREATE TABLE STARTPOINT_TEST (gid integer, ln1 db2gse.ST_LineString)
```

Las siguientes sentencias INSERT insertan las líneas en la columna LN1. La primera línea no tiene coordenadas Z ni medidas, mientras que la segunda línea tiene ambas cosas.

```
INSERT INTO STARTPOINT_TEST VALUES(1,
db2gse.ST_LineFromText('linestring (10.02 20.01,23.73
21.92,30.10 40.23)', db2gse.coordref()..srid(0)))

INSERT INTO STARTPOINT_TEST VALUES(2,
db2gse.ST_LineFromText('linestring  zm (10.02 20.01 5.0 7.0,
    23.73 21.92 6.5 7.1,30.10 40.23 6.9 7.2)',
                                db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente muestran cómo la función ST_StartPoint extrae el primer punto de cada línea. La función ST_AsText convierte el punto en su formato de texto. El primer punto de la lista no tiene coordenada Z ni medida, mientras que el segundo punto tiene ambas porque la línea fuente las tiene.

```
SELECT gid, CAST(db2gse.ST_AsText(db2gse.ST_StartPoint (ln1)) as varchar(60))
"Startpoint"
FROM STARTPOINT_TEST

GID         Startpoint
----------- ------------------------------------------------------------
          1 POINT ( 10.02000000 20.01000000)
          2 POINT ZM ( 10.02000000 20.01000000 5.00000000 7.00000000)

  2 record(s) selected.
```

ST_SymmetricDiff

ST_SymmetricDiff utiliza como entrada dos objetos de geometría y devuelve un objeto de geometría que es la diferencia simétrica de los objetos fuente.

La función ST_SymmetricDiff devuelve la diferencia simétrica (el operador lógico XOR para el espacio) de dos geometrías que se interseccionan y que tienen la misma dimensión. Si estas geometrías son iguales, ST_SymmetricDiff devuelve una geometría vacía. Si las geometrías no son iguales, una porción de una o ambas de ellas estará fuera de la zona de intersección. ST_SymmetricDiff devuelve la porción o porciones de no intersección en forma de conjunto de geometrías; por ejemplo, en forma de multipolígono.

Si las geometrías de entrada de ST_SymmetricDiff tienen dimensiones diferentes, la función devuelve un nulo.

Sintaxis

db2gse.ST_SymmetricDiff(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

El supervisor del condado debe determinar el área correspondiente a áreas sensibles que no forman intersección con el radio de cinco millas de sitios peligrosos.

La siguiente sentencia CREATE TABLE crea la tabla SENSITIVE_AREAS, que contiene varias columnas que describen las instituciones amenazadas. La tabla SENSITIVE_AREAS también contiene la columna ZONE, que guarda la geometría de polígono de cada institución.

```
CREATE TABLE SENSITIVE_AREAS (id        integer,
                              name      varchar(128),
                              size      float,
                              type      varchar(10),
                              zone      db2gse.ST_Polygon);
```

La siguiente sentencia CREATE TABLE crea la tabla HAZARDOUS_SITES, la cual almacena el identificador de los sitios en las columnas SITE_ID y NAME, mientras que la ubicación geográfica real de cada sitio se almacena en la columna LOCATION, de tipo punto.

```
CREATE TABLE HAZARDOUS_SITES (site_id   integer,
                              name      varchar(128),
                              location  point);
```

La función ST_Buffer genera un área de un radio de cinco millas que rodea las ubicaciones de los sitios de residuos peligrosos. La función ST_SymmetricDiff genera polígonos a partir de la intersección entre los

polígonos de sitios de residuos peligrosos rodeados y las áreas sensibles. La función ST_Area devuelve el área de los polígonos de la intersección correspondiente a cada sitio peligroso.

```
SELECT sa.name, hs.name,
       db2gse.ST_Area(db2gse.ST_SymmetricDiff (db2gse.ST_Buffer(hs.location,
(5 * 5280)),sa.zone))
FROM HAZARDOUS_SITES hs, SENSITIVE_AREAS sa
```

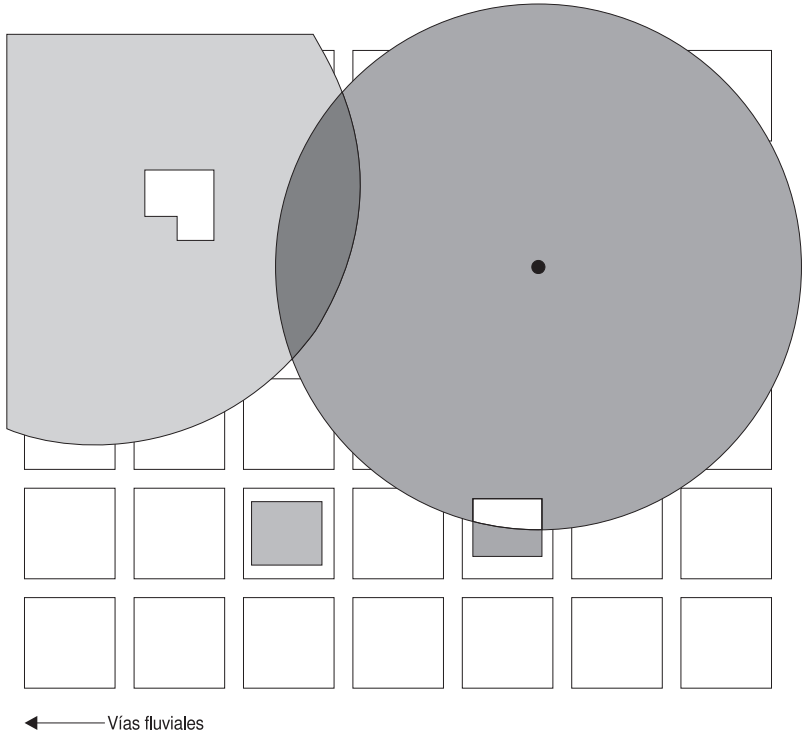


Figura 38. Utilización de ST_SymmetricDiff para determinar las áreas de residuos peligrosos que no contienen áreas sensibles (edificios habitados)

En la Figura 38, la diferencia simétrica de los sitios de residuos peligrosos y las áreas sensibles da como resultado la resta de las áreas que forman la intersección.

ST_Touches

ST_Touches devuelve un 1 (TRUE) si ninguno de los puntos comunes a ambas geometrías forma intersección con los interiores de ambas geometrías; si no es así, devuelve un 0 (FALSE). Al menos una geometría debe ser una línea, un polígono, una multilínea o un multipolígono.

Sintaxis

db2gse.ST_Touches(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

El técnico de GIS tiene que confeccionar una lista de líneas de alcantarillado cuyo final forma intersección con otra línea de alcantarillado.

La siguiente sentencia CREATE TABLE crea la tabla SEWERLINES, que tiene tres columnas. La primera columna, SEWER_ID, identifica de forma exclusiva cada línea de alcantarillado. La segunda columna, CLASS, de tipo entero, identifica el tipo de línea de alcantarillado, que suele estar asociado a la capacidad de la línea. La tercera columna, SEWER, de tipo línea, guarda la geometría de la línea de alcantarillado.

```
CREATE TABLE SEWERLINES (sewer_id integer, class integer, sewer
    db2gse.ST_LineString);
```

La siguiente sentencia SELECT devuelve una lista ordenada de SEWER_ID que se tocan entre sí.

```
SELECT s1.sewer_id, s2.sewer_id
FROM sewerlines s1, sewerlines s2
WHERE db2gse.ST_Touches (s1.sewer, s2.sewer) = 1,
ORDER BY 1,2;
```

ST_Transform

ST_Transform asocia una geometría a un sistema de referencias espaciales distinto del que tiene asignado actualmente la geometría.

Sintaxis

db2gse.ST_Transform(g db2gse.ST_Geometry, SRID db2gse.coordref)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla TRANSFORM_TEST, que tiene dos columnas tipo línea, L1 y L2.

```
CREATE TABLE TRANSFORM_TEST (tid integer, l1 db2gse.ST_LineString,
l2 db2gse.ST_LineString)
```

La siguiente sentencia INSERT inserta una línea en l1 con un SRID igual a 102.

```
INSERT INTO TRANSFORM_TEST VALUES (1, db2gse.ST_LineFromText('linestring
    (10.01 40.43, 92.32 29.89)',
    db2gse.coordref()..srid(102)),NULL)
```

La función ST_Transform convierte la línea de L1 de la referencia de coordenadas asignada a SRID 102 a la referencia de coordenadas asignada a SRID 105. La siguiente sentencia UPDATE guarda la línea transformada en la columna l2.

```
UPDATE TRANSFORM_TEST SET l2 = db2gse.ST_Transform(l1,
    db2gse.coordref()..srid(105))
```

Si la geometría asignada queda fuera del sistema de coordenadas asociado al nuevo sistema de referencias espaciales, ST_Transform devuelve un nulo.

Por ejemplo, considere una geometría ST_Point cuya coordenada X es 10.01 y cuya coordenada Y es 20.02. Suponga que esta geometría tiene asignado un sistema de referencias espaciales cuyos parámetros son:

falsex	0
falsey	0
xyunits	1

Ahora suponga que invoca ST_Transform para cambiar los parámetros del sistema de referencias espaciales de la geometría ST_Point por los parámetros siguientes:

falsex	100
falsey	100
xyunits	1

ST_Transform devolverá una geometría nula, pues las coordenadas (10.01, 20.02) quedan fuera del sistema de coordenadas asociado al nuevo sistema de referencias espaciales.

ST_Union

ST_Union utiliza como entrada dos objetos de geometría y devuelve un objeto de geometría que es la unión de los objetos fuente.

Sintaxis

db2gse.ST_Union(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla SENSITIVE_AREAS, que contiene varias columnas que describen las instituciones amenazadas. La tabla SENSITIVE_AREAS también contiene la columna ZONE, que guarda la geometría de polígono de cada institución.

```
CREATE TABLE SENSITIVE_AREAS (id       integer,
                              name     varchar(128),
                              size     float,
                              type     varchar(10),
                              zone     db2gse.ST_Polygon);
```

La siguiente sentencia CREATE TABLE crea la tabla HAZARDOUS_SITES, que almacena el identificador de los sitios en las columnas SITE_ID y NAME. La ubicación geográfica real de cada sitio se guarda en la columna LOCATION, de tipo punto.

```
CREATE TABLE HAZARDOUS_SITES (site_id integer, name varchar(128),
location db2gse.ST_Point);
```

La siguiente sentencia SELECT utiliza la función ST_Buffer para rodear con un radio de cinco millas las ubicaciones de sitios de residuos peligrosos. La función ST_Union genera polígonos a partir de la unión de los polígonos rodeados de sitios de residuos peligrosos y las áreas sensibles. La función ST_Area devuelve la unión de las áreas de los polígonos.

```
SELECT sa.name, hs.name,
       db2gse.ST_Area(db2gse.ST_Union(db2gse.ST_Buffer(hs.location,
(5 * 5280)),sa.zone))
FROM HAZARDOUS_SITES hs, SENSITIVE_AREAS sa;
```

ST_Within

ST_Within utiliza como entrada dos objetos de geometría y devuelve un 1 (TRUE) si el primer objeto queda completamente dentro del segundo y 0 (FALSE) si no es así.

Sintaxis

db2gse.ST_Within(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Tipo devuelto

Integer

Ejemplos

En el ejemplo siguiente, se crean dos tablas. La primera tabla, BUILDINGFOOTPRINTS, contiene las zonas edificadas de una ciudad. La segunda tabla, LOTS, contiene las parcelas de la ciudad. El ingeniero municipal desea asegurarse de que todas las áreas edificadas quedan completamente dentro de sus parcelas.

En ambas tablas, el tipo de datos multipolígono guarda la geometría de las zonas edificadas y sus parcelas. El diseñador de la base de datos ha seleccionado multipolígonos para ambos elementos geográficos, pues las parcelas pueden estar separadas por elementos naturales, tales como un río, y las zonas edificadas pueden constar de varios edificios.

```
CREATE TABLE BUILDINGFOOTPRINTS (   building_id integer,
                                    lot_id      integer,
                                    footprint   db2gse.ST_MultiPolygon);


CREATE TABLE LOTS (  lot_id integer, lot db2gse.ST_MultiPolygon );
```

Mediante la siguiente sentencia SELECT, el ingeniero municipal selecciona primero los edificios que no quedan completamente dentro de una parcela.

```
SELECT building_id
  FROM BUILDINGFOOTPRINTS, LOTS
 WHERE db2gse.ST_Within(footprint,lot) ≠ 1;
```

Aunque la primera consulta ofrecerá una lista de todos los BUILDING_ID que tienen zonas edificadas fuera de un polígono de parcela, no determinará si el resto tienen asignado el lot_id correcto. Esta segunda sentencia SELECT realiza una comprobación de integridad de los datos de la columna LOT_ID de la tabla BUILDINGFOOTPRINTS.

```
SELECT bf.building_id "building id",
       bf.lot_id "buildings lot_id",
       LOTS.lot_id "LOTS lot_id"
  FROM BUILDINGFOOTPRINTS bf, LOTS
 WHERE db2gse.ST_Within(footprint,lot) = 1 AND
       LOTS.lot_id <> bf.lot_id;
```

ST_WKBToSQL

ST_WKBToSQL genera un valor ST_Geometry a partir de su representación binaria convencional. El valor SRID de 0 se utiliza automáticamente.

Sintaxis

db2gse.ST_WKBToSQL(WKBGeometry Blob(1M))

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla LOTS, que tiene dos columnas: la columna LOT_ID, que identifica de forma exclusiva cada parcela, y la columna de multipolígono LOT, que contiene la geometría de cada parcela.

```
CREATE TABLE lots (lot_id  integer,
                   lot      db2gse.ST_MultiPolygon);
```

El siguiente fragmento de código C contiene funciones ODBC intercaladas con funciones SQL de Spatial Extender que insertan datos en la tabla LOTS.

La función ST_WKBToSQL convierte representaciones WKB en geometría de Spatial Extender. La sentencia INSERT completa se copia en una serie de caracteres wkb_sql. La sentencia INSERT contiene marcadores de parámetro para aceptar los datos LOT_ID y los datos LOT de forma dinámica.

```
/* Crear la sentencia insert de SQL para entrar datos
en el ID de parcela y el multipolígono de parcela. Los interrogantes son
marcadores de parámetro que indican el ID de parcela y los valores de
parcela que se recuperarán durante la ejecución. */

strcpy (wkb_sql,"insert into lots (lot_id, lot)
  values(?, db2gse.ST_WKBToSQL(cast(? as blob(1m))))");

/* Asignar memoria para el descriptor de sentencia SQL y
asociar el descriptor de sentencia con el descriptor de conexión.*/

rc = SQLAllocStmt (handle, &hstmt);

/* Preparar la sentencia SQL para la ejecución. */

rc = SQLPrepare (hstmt, (unsigned char *)wkb_sql, SQL_NTS);

/* Vincular el valor de clave de entero con el primer parámetro. */

pcbvalue1 = 0;
rc = SQLBindParameter (hstmt, 1, SQL_PARAM_INPUT, SQL_C_SLONG,
     SQL_INTEGER, 0, 0, &lot_id, 0, &pcbvalue1);
```

```
/* Vincular la forma con el segundo parámetro. */

pcbvalue2 = blob_len;
rc = SQLBindParameter (hstmt, 2, SQL_PARAM_INPUT, SQL_C_BINARY,
     SQL_BLOB, blob_len, 0, shape_blob, blob_len, &pcbvalue2);

/* Ejecutar la sentencia insert. */

rc = SQLExecute (hstmt);
```

ST_WKTToSQL

ST_WKTToSQL genera un valor ST_Geometry a partir de su representación de texto convencional. El valor SRID de 0 se utiliza automáticamente.

Sintaxis

db2gse.ST_WKTToSQL(geometryTaggedText Varchar(4000))

Tipo devuelto

db2gse.ST_Geometry

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla GEOMETRY_TEST que contiene dos columnas: la columna GID de tipo entero, que identifica de forma exclusiva cada fila, y la columna G1, que guarda la geometría.

```
CREATE TABLE GEOMETRY_TEST (gid smallint, g1 db2gse.ST_Geometry)
```

Las siguientes sentencias INSERT insertan los datos en las columnas GID y G1 de la tabla GEOMETRY_TEST. La función ST_WKTToSQL convierte la representación de texto de cada geometría en su correspondiente subclase replicable de Spatial Extender.

```
INSERT INTO GEOMETRY_TEST VALUES(
1, db2gse.ST_WKTToSQL ('point (10.02 20.01)')
)

INSERT INTO GEOMETRY_TEST VALUES(
2, db2gse.ST_WKTToSQL('linestring (10.01 20.01, 10.01 30.01, 10.01 40.01)')
)

INSERT INTO GEOMETRY_TEST VALUES(
3, db2gse.ST_WKTToSQL('polygon ((10.02 20.01, 11.92 35.64, 25.02 34.15,
19.15 33.94, 10.02 20.01))')
)

INSERT INTO GEOMETRY_TEST VALUES(
4, db2gse.ST_WKTToSQL('multipoint (10.02 20.01,10.32 23.98,11.92 25.64)')
)

INSERT INTO GEOMETRY_TEST VALUES(
5, db2gse.ST_WKTToSQL('multilinestring ((10.02 20.01,      10.32 23.98,
11.92 25.64),(9.55 23.75,15.36 30.11))')
)

INSERT INTO GEOMETRY_TEST VALUES(
6, db2gse.ST_WKTToSQL('multipolygon (((10.02 20.01, 11.92 35.64,
25.02 34.15, 19.15 33.94,10.02 20.01)),
((51.71 21.73, 73.36 27.04, 71.52 32.87, 52.43 31.90, 51.71 21.73)))')
)
```

ST_X

ST_X utiliza como entrada un punto y devuelve su coordenada X.

Sintaxis

ST_X(p ST_Point)

Tipo devuelto

Double

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla X_TEST, que tiene dos columnas: la columna GID, que identifica de forma exclusiva la fila, y la columna de tipo punto PT1.

```
CREATE TABLE X_TEST (gid integer, pt1 db2gse.ST_Point)
```

Las siguientes sentencias INSERT insertan dos filas. Una es un punto sin coordenada Z ni medida. La otra columna tiene una coordenada Z y una medida.

```
INSERT INTO X_TEST VALUES(1,
db2gse.ST_PointFromText('point (10.02 20.01)', db2gse.coordref()..srid(0)))

INSERT INTO X_TEST VALUES(2,
db2gse.ST_PointFromText('point  zm (10.02 20.01 5.0 7.0)',
    db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente listan la columna GID y la coordenada X doble de los puntos.

```
SELECT gid, db2gse.ST_X(pt1) "The X coordinate" FROM X_TEST

GID         The X coordinate
----------- ------------------------
          1   +1.00200000000000E+001
          2   +1.00200000000000E+001

  2 record(s) selected.
```

ST_Y

ST_Y utiliza como entrada un punto y devuelve su coordenada Y.

Sintaxis

db2gse.ST_Y(p db2gse.ST_Point)

Tipo devuelto

Double

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla Y_TEST, que tiene dos columnas: la columna GID, que identifica de forma exclusiva la fila, y la columna de tipo punto PT1.

```
CREATE TABLE Y_TEST (gid integer, pt1 db2gse.ST_Point)
```

Las sentencias INSERT insertan dos filas. Una es un punto sin coordenada Z ni medida. La otra columna tiene una coordenada Z y una medida.

```
INSERT INTO Y_TEST VALUES(1,
db2gse.ST_PointFromText('point (10.02 20.01)', db2gse.coordref()..srid(0)))

INSERT INTO Y_TEST VALUES(2,
db2gse.ST_PointFromText('point  zm (10.02 20.01 5.0 7.0)',
    db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente listan la columna GID y la coordenada Y doble de los puntos.

```
SELECT gid, db2gse.ST_Y(pt1) "The Y coordinate" FROM Y_TEST

GID         The Y coordinate
----------- ------------------------
          1   +2.00100000000000E+001
          2   +2.00100000000000E+001

  2 record(s) selected.
```

Z

Z utiliza como entrada un punto y devuelve su coordenada Z.

Sintaxis

Z(p db2gse.ST_Point)

Tipo devuelto

Double

Ejemplos

La siguiente sentencia CREATE TABLE crea la tabla Z_TEST con dos columnas: la columna GID, que identifica de forma exclusiva la fila, y la columna de tipo punto PT1.

```
CREATE TABLE Z_TEST (gid integer, pt1 db2gse.ST_Point)
```

Las siguientes sentencias INSERT insertan dos filas. Una es un punto sin coordenada Z ni medida. La otra columna tiene una coordenada Z y una medida.

```
INSERT INTO Z_TEST VALUES(1,
db2gse.ST_PointFromText('point (10.02 20.01)', db2gse.coordref()..srid(0)))

INSERT INTO Z_TEST VALUES(2,
db2gse.ST_PointFromText('point  zm (10.02 20.01 5.0 7.0)',
    db2gse.coordref()..srid(0)))
```

La siguiente sentencia SELECT y el conjunto resultante correspondiente listan la columna GID y la coordenada Z doble de los puntos. La primera fila es NULA porque el punto no tiene coordenada Z.

```
SELECT gid, db2gse.Z(pt1) "The Z coordinate" FROM Z_TEST

GID         The Z coordinate
----------- ------------------------
          1                        -
          2   +5.00000000000000E+000

  2 record(s) selected.
```

Capítulo 15. Sistemas de coordenadas

Este capítulo proporciona información de consulta sobre el sistema de referencias espaciales (Spatial Reference System, SRS) y los valores de coordenadas que se utilizan para interpretar datos espaciales.

Visión general de los sistemas de coordenadas

La representación de texto convencional de sistemas de referencias espaciales ofrece una representación textual estándar para la información de los sistemas de referencias espaciales. Las definiciones de la representación de texto convencional se basan en el modelo de datos de sistema de coordenadas POSC/EPSG (Petrotechnical Open Software Corporation/European Professional Surveyors Group).

Un sistema de referencias espaciales es un sistema de coordenadas geográficas (latitud-longitud), proyectadas (X,Y) o geocéntricas (X,Y,Z). El sistema de coordenadas consta de varios objetos. Cada objeto tiene una palabra clave en mayúsculas (por ejemplo, DATUM o UNIT) seguida de los parámetros separados por comas que definen el objeto entre paréntesis. Algunos objetos están compuestos por otros objetos, por lo que el resultado es una estructura anidada.

Nota: Las implantaciones pueden sustituir los paréntesis estándares ( ) por corchetes [ ] y deben estar preparadas para leer ambos formatos de paréntesis.

La definición de EBNF (Extended Backus Naur Form) correspondiente a la representación de series de un sistema de coordenadas que utiliza corchetes es la siguiente (consulte la nota anterior sobre el uso de paréntesis):

```
<sistema coordenadas> = <sc proyectadas> | <sc geográficas> | <sc geocéntricas>
<sc proyectadas> = PROJCS["<nombre>", <sc geográficas>, <proyección>,
    {<parámetro>,}* <unidad lineal>]
<proyección> = PROJECTION["<nombre>"]
<parámetro> = PARAMETER["<nombre>", <valor>]
<valor> = <número>
```

Un sistema de coordenadas de un conjunto de datos se identifica mediante la palabra clave PROJCS si los datos están en coordenadas proyectadas (mediante GEOGCS si están en coordenadas geográficas o mediante GEOCCS si están en coordenadas geocéntricas). La palabra clave PROJCS va seguida de todas las ″piezas″ que definen el sistema de coordenadas proyectadas. La primera pieza de cualquier objeto es siempre el nombre. Tras el nombre del sistema de coordenadas proyectadas hay varios objetos: el sistema de coordenadas geográficas, la proyección cartográfica, uno o más parámetros y la unidad lineal de medida. Todos los sistemas de coordenadas proyectadas se basan en un sistema de coordenadas geográficas, de modo que esta sección describe primero las piezas específicas de un sistema de coordenadas proyectadas. Por ejemplo, la zona UTM 10N del dato NAD83 se define del siguiente modo:

```
PROJCS["NAD_1983_UTM_Zone_10N",
<geographic cs>,
PROJECTION["Transverse_Mercator"],
PARAMETER["False_Easting",500000.0],
PARAMETER["False_Northing",0.0],
PARAMETER["Central_Meridian",-123.0],
PARAMETER["Scale_Factor",0.9996],
PARAMETER["Latitude_of_Origin",0.0],
UNIT["Meter",1.0]]
```

El nombre y varios objetos definen el objeto sistema de coordenadas geográficas: el dato, el meridiano principal y la unidad angular de medida.

```
<sc geográficas> = GEOGCS["<nombre>", <dato>, <meridiano principal>,
    <unidad angular>]
<dato> = DATUM["<nombre>", <esferoide>]
<esferoide> = SPHEROID["<nombre>", <eje semi-principal>, <allanamiento inverso>]
<eje semi-principal> = <número>
    (eje semi-principal se mide en metros y debe ser > 0.)
<allanamiento inverso> = <número>
<meridiano principal> = PRIMEM["<nombre>", <longitud>]
<longitud> = <número>
```

La serie del sistema de coordenadas geográficas para la zona UTM 10 en NAD83:

```
GEOGCS["GCS_North_American_1983",
DATUM["D_North_American_1983",
SPHEROID["GRS_1980",6378137,298.257222101]],
PRIMEM["Greenwich",0],
UNIT["Degree",0.0174532925199433]]
```

El objeto UNIT puede representar unidades de medida angulares o lineales:

```
<unidad angular> = <unidad>
<unidad lineal> = <unidad>
<unidad> = UNIT["<nombre>", <factor conversión>]
<factor conversión> = <número>
```

El factor de conversión especifica número de metros (para una unidad lineal) o número de radianes (para una unidad angular) por unidad y debe ser mayor que cero.

De este modo, la representación completa de la serie de Zona UTM 10N es la siguiente:

```
PROJCS["NAD_1983_UTM_Zone_10N",
GEOGCS["GCS_North_American_1983",
DATUM[ "D_North_American_1983",SPHEROID["GRS_1980",6378137,298.257222101]],
PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925199433]],
PROJECTION["Transverse_Mercator"],PARAMETER["False_Easting",500000.0],
PARAMETER["False_Northing",0.0],PARAMETER["Central_Meridian",-123.0],
PARAMETER["Scale_Factor",0.9996],PARAMETER["Latitude_of_Origin",0.0],
UNIT["Meter",1.0]]
```

Un sistema de coordenadas geocéntricas es bastante parecido a un sistema de coordenadas geográficas:

```
<sc geocéntricas> = GEOCCS["<nombre>", <dato>, <meridiano principal>,
    <unidad lineal>]
```

Unidades lineales soportadas

Tabla 58. Unidades lineales soportadas

Unidad	Factor de conversión
Metro	1,0
Pie (Internacional)	0,3048
Pie EE.UU.	12/39,37
Pie americano modificado	12,0004584/39,37
Pie de Clarke	12/39,370432
Pie Indian	12/39,370141
Link	7,92/39,370432
Link (Benoit)	7,92/39,370113
Link (Sears)	7,92/39,370147
Chain (Benoit)	792/39,370113
Chain (Sears)	792/39,370147
Yarda (Indian)	36/39,370141

Tabla 58. Unidades lineales soportadas (continuación)

Unidad	Factor de conversión
Yarda (Sears)	36/39,370147
Braza	1,8288
Milla náutica	1852,0

Unidades angulares soportadas

Tabla 59. Unidades angulares soportadas

Unidad	Factor de conversión
Radián	1,0
Grado decimal	$\pi/180$
Minuto decimal	$(\pi/180)/60$
Segundo decimal	$(\pi/180)/36000$
Gon	$\pi/200$
Grado centesimal	$\pi/200$

Esferoides soportados

Tabla 60. Esferoides soportados

Nombre	Eje semi-principal	Allanamiento inverso
Airy	6377563,396	299,3249646
Modified Airy	6377340,189	299,3249646
Australian	6378160	298,25
Bessel	6377397,155	299,1528128
Modified Bessel	6377492,018	299,1528128
Bessel (Namibia)	6377483,865	299,1528128
Clarke 1866	6378206,4	294,9786982
Clarke 1866 (Michigan)	6378693,704	294,978684677
Clarke 1880	6378249,145	293,465
Clarke 1880 (Arc)	6378249,145	293,466307656
Clarke 1880 (Benoit)	6378300,79	293,466234571
Clarke 1880 (IGN)	6378249,2	293,46602
Clarke 1880 (RGS)	6378249,145	293,465

Tabla 60. Esferoides soportados (continuación)

Nombre	Eje semi-principal	Allanamiento inverso
Clarke 1880 (SGA)	6378249,2	293,46598
Everest 1830	6377276,345	300,8017
Everest 1975	6377301,243	300,8017
Everest (Sarawak y Sabah)	6377298,556	300,8017
Modified Everest 1948	6377304,063	300,8017
Fischer 1960	6378166	298,3
Fischer 1968	6378150	298,3
Modified Fischer (1960)	6378155	298,3
GEM10C	6378137	298,257222101
GRS 1980	6378137	298,257222101
Hayford 1909	6378388	297,0
Helmert 1906	6378200	298,3
Hough	6378270	297,0
Internacional 1909	6378388	297,0
Internacional 1924	6378388	297,0
Nuevo internacional 1967	6378157,5	298,2496
Krasovsky	6378245	298,3
Mercury 1960	6378166	298,3
Modified Mercury 1968	6378150	298,3
NWL9D	6378145	298,25
OSU_86F	6378136,2	298,25722
OSU_91A	6378136,3	298,25722
Plessis 1817	6376523	308,64
South American 1969	6378160	298,25
Southeast Asia	6378155	298,3
Sphere (radius = 1.0)	1	0
Sphere (radius = 6371000 m)	6371000	0
Sphere (radius = 6370997 m)	6370997	0
Struve 1860	6378297	294,73
Walbeck	6376896	302,78
War Office	6378300,583	296

Tabla 60. Esferoides soportados (continuación)

Nombre	Eje semi-principal	Allanamiento inverso
WGS 1960	6378165	298,3
WGS 1966	6378145	298,25
WGS 1972	6378135	298,26
WGS 1984	6378137	298,257223563

Datos geodésicos soportados

Tabla 61. Datos geodésicos soportados

Datos geodésicos soportados	
Adindan	Lisbon
Afgooye	Loma Quintana
Agadez	Lome
Australian Geodetic Datum 1966	Luzon 1911
Australian Geodetic Datum 1984	Mahe 1971
Ain el Abd 1970	Makassar
Amersfoort	Malongo 1987
Aratu	Manoca
Arc 1950	Massawa
Arc 1960	Merchich
Ancienne Triangulation Francaise	Militar-Geographische Institute
Barbados	Mhast
Batavia	Minna
Beduaram	Monte Mario
Beijing 1954	M'poraloko
Reseau National Belge 1950	NAD Michigan
Reseau National Belge 1972	North American Datum 1927
Bermuda 1957	North American Datum 1983
Bern 1898	Nahrwan 1967
Bern 1938	Naparima 1972
Bogota	Nord de Guerre
Bukit Rimpah	NGO 1948
Camacupa	Nord Sahara 1959

Tabla 61. Datos geodésicos soportados (continuación)

Datos geodésicos soportados	
Campo Inchauspe	NSWC 9Z-2
Cape	Nouvelle Triangulation Francaise
Carthage	New Zealand Geodetic Datum 1949
Chua	OS (SN) 1980
Conakry 1905	OSGB 1936
Corrego Alegre	OSGB 1970 (SN)
Cote d'Ivoire	Padang 1884
Datum 73	Palestine 1923
Deir ez Zor	Pointe Noire
Deutsche Hauptdreiecksnetz	Provisional South American Datum 1956
Douala	Pulkovo 1942
European Datum 1950	Qatar
European Datum 1987	Qatar 1948
Egypt 1907	Qornoq
European Reference System 1989	RT38
Fahud	South American Datum 1969
Gandajika 1970	Sapper Hill 1943
Garoua	Schwarzeck
Geocentric Datum of Australia 1994	Segora
Guyane Francaise	Serindung
Herat North	Stockholm 1938
Hito XVIII 1963	Sudan
Hu Tzu	Shan Tananarive 1925
Hungarian Datum 1972	Timbalai 1948
Indian 1954	TM65
Indian 1975	TM75
Indonesian Datum 1974	Tokyo
Jamaica 1875	Trinidad 1903
Jamaica 1969	Trucial Coast 1948
Kalianpur	Voirol 1875
Kandawala	Voirol Unifie 1960
Kertau	WGS 1972

Tabla 61. Datos geodésicos soportados (continuación)

Datos geodésicos soportados	
Kuwait Oil Company	WGS 1972 Transit Broadcast Ephemeris
La Canoa	WGS 1984
Lake	Yacare
Leigon	Yoff
Liberia 1964	Zanderij

Meridianos principales soportados

Tabla 62. Meridianos principales soportados

Ubicación	**Coordenadas**
Greenwich	0° 0' 0"
Berna	7° 26' 22,5" E
Bogota	74° 4' 51,3" W
Bruselas	4° 22' 4,71" E
Ferro	17° 40' 0" W
Yakarta	106° 48' 27,79" E
Lisboa	9° 7' 54,862" W
Madrid	3° 41' 16,58" W
París	2° 20' 14,025" E
Roma	12° 27' 8,4" E
Estocolmo	18° 3' 29" E

Proyecciones cartográficas soportadas

Tabla 63. Proyecciones cartográficas soportadas

Proyecciones cilíndricas	**Proyecciones seudocilíndricas**
Behrmann	Parabólica de Craster
Cassini	Eckert I
Área igual cilíndrica	Eckert II
Equirectangular	Eckert III
Estereográfica de Gall	Eckert IV
Gauss-Kruger	Eckert V

Tabla 63. Proyecciones cartográficas soportadas (continuación)

Proyecciones cilíndricas	**Proyecciones seudocilíndricas**
Mercator	Eckert VI
Cilíndrica de Miller	Cuártico polar plano de McBryde-Thomas
Oblicua	Mercator (Hotine) Mollweide
Plate-Carée	Robinson
Times	Sinusoidal (Sansom-Flamsteed)
Transverse Mercator	Winkel I

Proyecciones cónicas soportadas

Tabla 64. Proyecciones cónicas soportadas

Proyecciones cónicas soportadas	
Equivalente cónica de Albers	Trimétrica de Chamberlin
Cónica conforme oblicua bipolar	Equidistante a dos puntos
Bonne	Equivalente de Hammer-Aitoff
Cónica equidistante	Van der Grinten I
Cónica conforme Lambert	Variada
Policónica	Alaska serie E
Cónica simple	Alaska Grid (Estereográfica modificada por Snyder)

Proyecciones azimutales o planares soportadas

- Equidistante azimutal
- Perspectiva general lado interior vertical
- Nomónica
- Equivalente azimuthal de Lambert
- Ortogonal
- Estereográfica polar
- Estereográfica

Parámetros de proyección cartográfica soportados

Tabla 65. Parámetros de proyección cartográfica soportados

Parámetro	Descripción
central_meridian	Línea de longitud geográfica elegida como origen de las coordenadas x.
scale_factor	Se utiliza generalmente para reducir la cantidad de distorsión en una proyección cartográfica.
standard_parallel_1	Línea de latitud sin distorsión importante. También se utiliza para la ″latitud de escala real″.
standard_parallel_2	Línea de longitud geográfica sin distorsión importante.
longitude_of_center	Longitud geográfica que define el punto central de la proyección cartográfica.
latitude_of_center	Latitud que define el punto central de la proyección cartográfica.
longitude_of_origin	Longitud geográfica elegida como origen de las coordenadas X.
latitude_of_origin	Latitud elegida como origen de las coordenadas Y.
false_easting	Se suma a las coordenadas X. Sirve para proporcionar valores positivos.
false_northing	Se suma a las coordenadas Y. Sirve para proporcionar valores positivos.
azimuth	Ángulo al este del norte que define la línea central de una proyección oblicua.
longitude_of_point_1	Longitud geográfica del primer punto necesario para una proyección cartográfica.
latitude_of_point_1	Latitud del primer punto necesario para una proyección cartográfica.
longitude_of_point_2	Longitud geográfica del segundo punto necesario para una proyección cartográfica.
latitude_of_point_2	Latitud del segundo punto necesario para una proyección cartográfica.
longitude_of_point_3	Longitud geográfica del tercer punto necesario para una proyección cartográfica.

Tabla 65. Parámetros de proyección cartográfica soportados (continuación)

Parámetro	Descripción
latitude_of_point_3	Latitud del tercer punto necesario para una proyección cartográfica.
landsat_number	Número de un satélite Landsat.
path_number	Número de ruta orbital de un determinado satélite.
perspective_point_height	Altura por encima de la superficie terrestre del punto de perspectiva de la proyección cartográfica.
fipszone	Número de zona del Sistema de coordenadas de plano de estado.
zone	Número de zona UTM.

Capítulo 16. Formatos de archivo para datos espaciales

Este capítulo documenta las representaciones convencionales de Spatial Extender. Las representaciones se describen como *convencionales* porque están definidas por OGC (Open GIS Consortium) y no son específicas de Spatial Extender. Existen tres tipos de valores espaciales que son importantes para comprender el proceso de importación y exportación de datos espaciales:

- Las representaciones de texto convencionales (representaciones WKT) de OGC
- Las representaciones binarias convencionales (representaciones WKB) de OGC
- Las representaciones de forma de ESRI

Representaciones de texto convencionales de OGC

Spatial Extender tiene varias funciones que generan geometrías a partir de descripciones de texto:

ST_GeomFromText
Crea una geometría a partir de una representación de texto de cualquier tipo de geometría.

ST_PointFromText
Crea un punto a partir de una representación de texto de un punto.

ST_LineFromText
Crea una línea a partir de una representación de texto de una línea.

ST_PolyFromText
Crea un polígono a partir de una representación de texto de un polígono.

ST_MPointFromText
Crea un multipunto a partir de una representación de texto de un multipunto.

ST_MLineFromText
Crea una multilínea a partir de una representación de texto de una multilínea.

ST_MPolyFromText
Crea un multipolígono a partir de una representación de texto de un multipolígono.

La representación de texto es una serie de caracteres en formato ASCII que permite intercambiar una geometría en formato de texto ASCII. Puede utilizar estas funciones en un programa en lenguaje de tercera o cuarta generación (3GL o 4GL), pues no es necesario definir ninguna estructura de programa especial. La función ST_AsText convierte una geometría en una representación de texto.

Cada tipo de geometría tiene una representación de texto convencional, que se puede utilizar para crear nuevas instancias del tipo y para convertir las instancias existentes a formato textual para su visualización alfanumérica.

La representación de texto convencional de una geometría se define del siguiente modo: la notación {}* indica 0 o más repeticiones de los símbolos encerrados entre llaves; las llaves no aparecen en la lista de símbolos de salida.

```
<Geometry Tagged Text> :=
| <Point Tagged Text>
| <LineString Tagged Text>
| <Polygon Tagged Text>
| <MultiPoint Tagged  Text>
| <MultiLineString Tagged Text>
| <MultiPolygon Tagged Text>

<Point Tagged Text> :=
POINT (<Point Text>)

<LineString Tagged Text> :=
LINESTRING (<LineString Text>)

<Polygon Tagged Text> :=
POLYGON (<Polygon Text>)

<MultiPoint Tagged Text> :=
MULTIPOINT (<MultiPoint Text>)

<MultiLineString Tagged Text> :=
MULTILINESTRING (<MultiLineString Text>)

<MultiPolygon Tagged Text> :=
MULTIPOLYGON (<MultiPolygon Text>)

<Point Text> := EMPTY
|    <Point>
| Z  (<PointZ>)
| M  (<PointM>)
| ZM (<PointZM>)

<Point> :=  <x>  <y>
<x> := literal de doble precisión
<y> := literal de doble precisión
<PointZ> :=  <x>  <y>  <z>
<x> := literal de doble precisión
```

```
<y> := literal de doble precisión
<z> := literal de doble precisión
<PointM> :=  <x>  <y>  <m>
<x> := literal de doble precisión
<y> := literal de doble precisión
<m> := literal de doble precisión
<PointZM> :=  <x>  <y>  <z>  <m>
<x> := literal de doble precisión
<y> := literal de doble precisión
<z> := literal de doble precisión
<m> := literal de doble precisión

<LineString Text> := EMPTY
|    ( <Point Text >   {,  <Point Text> }*  )
| Z  ( <PointZ Text >  {,  <PointZ Text> }*  )
| M  ( <PointM Text >  {,  <PointM Text> }*  )
| ZM ( <PointZM Text > {,  <PointZM Text> }*  )

<Polygon Text> := EMPTY
| ( <LineString Text > {,< LineString Text > }*  )

<MultiPoint Text> := EMPTY
| ( <Point Text >   {,  <Point Text > }*  )

<MultiLineString Text> := EMPTY
| ( <LineString Text > {,< LineString Text>}*  )

<MultiPolygon Text> := EMPTY
| ( < Polygon Text > {,  < Polygon Text > }*  )
```

La sintaxis básica de la función es:

```
function (<texto descriptivo>,<SRID db2gse.coordref>)
```

El SRID, el identificador de referencias espaciales y clave principal de la tabla SPATIAL_REFERENCES, identifica el sistema de referencias espaciales de la geometría que se almacena en la tabla SPATIAL_REFERENCES. Para que una geometría se pueda insertar en una columna espacial, su SRID debe coincidir con el SRID de la columna espacial.

El texto descriptivo está formado por tres componentes básicos que se encierran entre comillas simples, por ejemplo:

```
<'tipo de geometría'> ['tipo de coordenadas'] [''lista de coordenadas']
```

donde:

tipo de geometría
: Es uno de los siguientes: punto, línea, polígono, multipunto, multilínea o multipolígono.

tipo de coordenadas
: Especifica si la geometría tiene o no coordenadas Z o medidas. Deje este argumento en blanco si la geometría no tiene ni coordenadas Z ni

medidas. Si no es así, defina el tipo de coordenadas como Z para las geometrías que contienen coordenadas Z, M para las geometrías con medidas y ZM para las geometrías con ambas.

lista de coordenadas
Define los vértices de la geometría. Las listas de coordenadas están delimitadas por comas y encerradas entre paréntesis. Las geometrías con varios componentes necesitan grupos de paréntesis para encerrar cada parte del componente. Si la geometría está vacía, la palabra clave EMPTY sustituye a la coordenada.

La Tabla 66 muestra una lista de ejemplos de todos los posibles textos descriptivos.

Tabla 66. Tipos de geometría y sus textos descriptivos

Tipo de geometría	Texto descriptivo	Comentario
punto	point empty	punto vacío
punto	point z empty	punto vacío con coordenada z
punto	point m empty	punto vacío con medida
punto	point zm empty	punto vacío con coordenada z y medida
punto	point (10.05 10.28)	punto
punto	point z (10.05 10.28 2.51)	punto con coordenada z
punto	point m (10.05 10.28 4.72)	punto con medida
punto	point zm (10.05 10.28 2.51 4.72)	punto con coordenada z y medida
línea	linestring empty	línea vacía
línea	linestring z empty	línea vacía con coordenadas z
línea	linestring m empty	línea vacía con medidas
línea	linestring zm empty	línea vacía con coordenadas z y medidas
línea	linestring (10.05 10.28 , 20.95 20.89)	línea
línea	linestring z (10.05 10.28 3.09, 20.95 31.98 4.72, 21.98 29.80 3.51)	línea con coordenadas z
línea	linestring m (10.05 10.28 5.84, 20.95 31.98 9.01, 21.98 29.80 12.84)	línea con medidas

Tabla 66. Tipos de geometría y sus textos descriptivos (continuación)

Tipo de geometría	Texto descriptivo	Comentario
línea	linestring zm ( )	línea con coordenadas z y medidas
polígono	polygon empty	polígono vacío
polígono	polygon z empty	polígono vacío con coordenadas z
polígono	polygon m empty	polígono vacío con medidas
polígono	polygon zm empty	polígono vacío con coordenadas z y medidas
polígono	polygon ((10 10, 10 20, 20 20, 20 15, 10 10))	polígono
polígono	polygon z (( ))	polígono con coordenadas z
polígono	polygon m (( ))	polígono con medidas
polígono	polygon zm (( ))	polígono con coordenadas z y medidas
multipunto	multipoint empty	multipunto vacío
multipunto	multipoint z empty	multipunto vacío con coordenadas z
multipunto	multipoint m empty	multipunto vacío con medidas
multipunto	multipoint zm empty	multipunto vacío con coordenadas z y medidas
multipunto	multipoint empty	multipunto vacío
multipunto	multipoint (10 10, 20 20)	multipunto con dos puntos
multipunto	multipoint z (10 10 2, 20 20 3)	multipunto con coordenadas z
multipunto	multipoint m (10 10 4, 20 20 5)	multipunto con medidas
multipunto	multipoint zm (10 10 2 4, 20 20 3 5)	multipunto con coordenadas z y medidas
multilínea	multilinestring empty	multilínea vacía
multilínea	multilinestring z empty	multilínea vacía con coordenadas z
multilínea	multilinestring m empty	multilínea vacía con medidas
multilínea	multilinestring zm empty	multilínea vacía con coordenadas z y medidas
multilínea	multilinestring (( ))	multilínea

Tabla 66. Tipos de geometría y sus textos descriptivos (continuación)

Tipo de geometría	Texto descriptivo	Comentario
multilínea	multilinestring z (( ))	multilínea con coordenadas z
multilínea	multilinestring m (( ))	multilínea con medidas
multilínea	multilinestring zm (( ))	multilínea con coordenadas z y medidas
multipolígono	multipolygon empty	multipolígono vacío
multipolígono	multipolygon z empty	multipolígono vacío con coordenadas z
multipolígono	multipolygon m empty	multipolígono vacío con medidas
multipolígono	multipolygon z	multipolígono vacío con coordenadas y medidas
multipolígono	multipolygon ((()))	multipolígono
multipolígono	multipolygon z ((( )))	multipolígono con coordenadas z
multipolígono	multipolygon m (((10 10 2, 10 20 3, 20 20 4, 20 15 5, 10 10 2), (50 40 7, 50 50 3, 60 50 4, 60 40 5, 50 40 7)))	multipolígono con medidas
multipolígono	multipolygon zm ((( )))	multipolígono con coordenadas z y medidas

Representaciones binarias convencionales (representaciones WKB) de OGC

Spatial Extender tiene varias funciones que generan geometrías a partir de representaciones binarias:

ST_GeomFromWKB
Crea una geometría a partir de una representación WKB de cualquier tipo de geometría.

ST_PointFromWKB
Crea un punto a partir de una representación WKB de punto.

ST_LineFromWKB
Crea una línea a partir de una representación WKB de línea.

ST_PolyFromWKB
Crea un polígono a partir de una representación WKB de polígono.

ST_MPointFromWKB
Crea un multipunto a partir de una representación WKB de multipunto.

ST_MLineFromWKB
Crea una multilínea a partir de una representación WKB de multilínea.

ST_MPolyFromWKB
Crea un multipolígono a partir de una representación WKB de multipolígono.

La representación binaria convencional es una corriente continua de bytes. Permite intercambiar una geometría entre un cliente ODBC y una base de datos SQL en formato binario. Puesto que estas funciones de geometría necesitan la definición de estructuras de lenguaje de programación C para correlacionar la representación binaria, están destinadas a ser utilizadas dentro de un programa de lenguaje de tercera generación (3GL). No se ajustan a un entorno de lenguaje de cuarta generación (4GL). La función ST_AsBinary convierte un valor de geometría existente en una representación binaria convencional.

La representación binaria convencional correspondiente a la geometría se obtiene serializando una instancia de la geometría como una secuencia de tipos numéricos. Estos tipos se extraen del grupo (entero sin signo, doble precisión), y luego cada tipo numérico se serializa como una secuencia de bytes. Los tipos se serializan utilizando una de las dos representaciones binarias definidas, estándar, para tipos numéricos (NDR, XDR). Un identificador de un byte que precede los bytes serializados describe el código binario específico (NDR o XDR) utilizado para una corriente de bytes de geometría. La única diferencia entre los dos códigos de geometría es el orden de los bytes: el código XDR es Big Endian; el código NDR es Little Endian.

Definiciones de tipos numéricos

Un *entero sin signo* es un tipo de datos de 32 bits (4 bytes) que codifica un entero no negativo dentro del rango [1, 4294967295].

Un *número doble* es un tipo de datos de doble precisión de 64 bits (8 bytes) que codifica un número de doble precisión utilizando el formato de doble precisión 754 del IEEE.

Estas definiciones son comunes a XDR y a NDR.

Codificación XDR (Big Endian) de tipos numéricos

La representación XDR de un entero sin signo es Big Endian (primero el byte más significativo).

La representación XDR de un número doble es Big Endian (el bit de signo es el primer bit).

Codificación NDR (Little Endian) de tipos numéricos

La representación NDR de un entero sin signo es Little Endian (primero el byte menos significativo).

La representación NDR de un número doble es Little Endian (el bit de signo es el último byte).

Conversión entre NDR y XDR

La conversión entre los tipos de datos NDR y XDR para enteros sin signo y números dobles es una operación sencilla. Esta operación comporta invertir el orden de los bytes dentro de cada entero sin signo o número doble de la corriente de bytes.

Descripción de las corrientes de bytes WKBGeometry

Esta sección describe la representación binaria convencional correspondiente a una geometría. El elemento básico es la corriente de bytes correspondiente a un punto, que consta de dos números de doble precisión. Las corrientes de bytes de otras geometrías se crean utilizando las corrientes de bytes correspondientes a geometrías que ya están definidas.

```
// Definiciones de los tipos básicos
// byte : 1 byte
// uint32 : entero sin signo de 32 bits (4 bytes)
// double : número de doble precisión (8 bytes)

// Componentes básicos : Point, LinearRing

Point {
  double x;
  double y;
};
LinearRing   {
  uint32   numPoints;
  Point    points[numPoints];
};
enum wkbGeometryType {
  wkbPoint = 1,
  wkbLineString = 2,
  wkbPolygon = 3,
  wkbMultiPoint = 4,
  wkbMultiLineString = 5,
  wkbMultiPolygon = 6
};
enum wkbByteOrder {
  wkbXDR = 0,                                    // Big Endian
  wkbNDR = 1                                   // Little Endian
};
WKBPoint {
  byte                byteOrder;
```

```
  uint32   wkbType;                                          // 1
  Point    point;
};
WKBLineString {
  byte                byteOrder;
  uint32   wkbType;                                          // 2
  uint32   numPoints;
  Point    points[numPoints];
};

WKBPolygon    {
  byte              byteOrder;
  uint32            wkbType;                                 // 3
  uint32            numRings;
  LinearRing        rings[numRings];
};
WKBMultiPoint    {
  byte              byteOrder;
  uint32            wkbType;                                 // 4
  uint32            num_wkbPoints;
  WKBPoint            WKBPoints[num_wkbPoints];
};
WKBMultiLineString    {
  byte              byteOrder;
  uint32            wkbType;                                 // 5
  uint32            num_wkbLineStrings;
  WKBLineString     WKBLineStrings[num_wkbLineStrings];
};

wkbMultiPolygon {
  byte              byteOrder;
  uint32            wkbType;                                 // 6
  uint32            num_wkbPolygons;
  WKBPolygon        wkbPolygons[num_wkbPolygons];
};

;WKBGeometry  {
  union {
    WKBPoint                point;
    WKBLineString           linestring;
    WKBPolygon              polygon;
    WKBMultiPoint           mpoint;
    WKBMultiLineString      mlinestring;
    WKBMultiPolygon         mpolygon;
  }
};
```

La siguiente figura muestra una representación NDR.

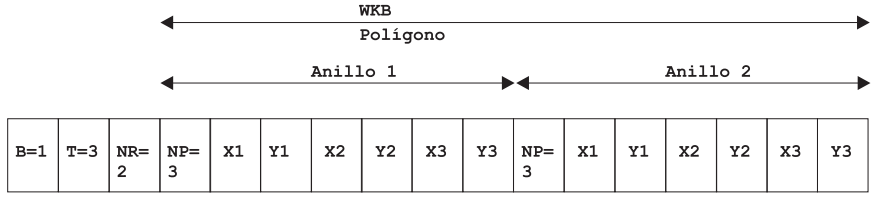


Figura 39. Representación en formato NDR. (B=1) de tipo polígono (T=3) con 2 lineales (NR=2), cada anillo tiene 3 puntos (NP=3).

Declaraciones sobre la representación WKB

La representación binaria convencional de una geometría está diseñada para representar instancias de los tipos de geometría descritos en el Modelo de objetos de geometría y en la Especificación de resúmenes técnicos de OpenGIS.

Estas especificaciones implican lo siguiente para anillos, polígonos y multipolígonos:

Anillos lineales
Los anillos son simples y cerrados, lo que significa que los anillos lineales no pueden formar intersección con ellos mismos.

Polígonos
En el límite de un polígono no puede haber dos anillos lineales que formen intersección entre sí. Los anillos lineales del límite de un polígono pueden formar intersección como máximo en un punto individual, pero sólo como tangente.

Multipolígonos
Los interiores de dos polígonos que son elementos de un multipolígono no pueden formar intersección. Los límites de dos polígonos cualesquiera que son elementos de un multipolígono pueden tener sólo un número finito de puntos de contacto.

Las representaciones de forma ESRI

Spatial Extender tiene varias funciones que generan geometrías a partir de representaciones de forma ESRI. Además de la representación bidimensional soportada por la representación binaria convencional de OGC, la representación de forma ESRI también da soporte a coordenadas Z y medidas opcionales. Las siguientes funciones generan una geometría a partir de una forma ESRI:

GeometryFromShape
Crea una geometría a partir de una representación de forma de cualquier tipo de geometría.

PointFromShape
Crea un punto a partir de una representación de forma de un punto.

LineFromShape
Crea una línea a partir de una representación de forma de una línea.

PolyFromShape
Crea un polígono a partir de una representación de forma de un polígono.

MPointFromShape
Crea un multipunto a partir de una representación de forma de un multipunto.

MLineFromShape
Crea una multilínea a partir de una representación de forma de una multilínea.

MPolyFromShape
Crea un multipolígono a partir de una representación de forma de un multipolígono.

La sintaxis general de estas funciones es la misma. El primer argumento es la representación de forma que se entra como tipo de datos objeto grande binario (BLOB). El segundo argumento es el entero identificador de referencias espaciales que se va a asignar a la geometría. La función GeometryFromShape tiene la siguiente sintaxis:

```
db2gse.GeometryFromShape(ShapeGeometry Blob(1M), cr db2gse.coordref)
```

Puesto que estas funciones de forma necesitan la definición de estructuras de lenguaje de programación C para correlacionar la representación binaria, están destinadas a ser utilizadas dentro de un programa 3GL y no se ajustan a un entorno 4GL. La función AsShape convierte el valor de geometría en una representación de forma ESRI.

El tipo de forma 0 denota una forma nula, sin datos geométricos para la forma.

Valor	Tipo de forma
0	Null Shape (forma nula)
1	Point (punto)
3*	PolyLine (polilínea)
5	Polygon (polígono)

Valor	Tipo de forma
8	MultiPoint (multipunto)
11	PointZ (puntoZ)
13	PolyLineZ (polilíneaZ)
15	PolygonZ (polígonoZ)
18	MultiPointZ (multipuntoZ)
21	PointM (puntoM)
23	PolyLineM (polilíneaM)
25	PolygonM (polígonoM)
28	MultiPointM (multipuntoM)

Nota: * Los tipos de formas no especificados en la tabla anterior (2, 4, 6, etc.) se reservan para un futuro uso.

Tipos de forma en el espacio XY

Punto

Un punto consta de un par de coordenadas de doble precisión en el orden X, Y.

Tabla 67. Contenido de la corriente de bytes de un punto

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	1	Integer	1	Little
Byte 4	X	X	Double	1	Little
Byte 12	Y	Y	Double	1	Little

Multipunto

Un multipunto consta de un grupo de puntos. El recuadro delimitador ("box") se guarda en el orden Xmín, Ymín, Xmáx, Ymáx.

Tabla 68. Contenido de la corriente de bytes de un multipunto

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	8	Integer	1	Little
Byte 4	Box	Box	Double	4	Little
Byte 36	NumPoints	NumPoints	Integer	1	Little
Byte 40	Points	Points	Point	NumPoints	Little

Polilínea

Una polilínea es un conjunto ordenado de vértices que consta de una o más partes. Una parte es una secuencia conectada de dos o más puntos. Los puntos pueden estar o no conectados entre sí. Las partes pueden formar o no formar intersección entre sí.

Puesto que esta especificación no prohíbe la existencia de puntos consecutivos con iguales coordenadas, los lectores de archivos de forma deben abordar esa posibilidad. En cambio, no se permiten las partes degeneradas de longitud cero que pueden derivar de esa posibilidad.

Los campos de una polilínea son:

Box Recuadro delimitador de la polilínea que se almacena en el orden Xmín, Ymín, Xmáx, Ymáx.

NumParts
Número de partes de la polilínea.

NumPoints
Número total de puntos correspondientes a todas las partes.

Parts Matriz de longitud NumParts. Cada polilínea almacena el índice de su primer punto en la matriz de puntos. Los índices de la matriz están referidos a 0.

Points Matriz de longitud NumPoints. Los puntos de cada parte de la polilínea se almacenan de forma consecutiva. Los puntos correspondientes a la parte 2 siguen a los correspondientes a la parte 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del punto inicial correspondiente a cada parte. No hay ningún delimitador en la matriz de puntos entre partes.

Tabla 69. Contenido de la corriente de bytes de una polilínea

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	3	Integer	1	Little
Byte 4	Box	Box	Double	4	Little
Byte 36	NumParts	NumParts	Integer	1	Little
Byte 40	NumPoints	NumPoints	Integer	1	Little
Byte 44	Parts	Parts	Integer	NumParts	Little
Byte X	Points	Points	Point	NumPoints	Little

Nota: X = 44 + 4 * NumParts.

Polygon

Un polígono consta de uno o más anillos. Un anillo es una secuencia conectada de cuatro o más puntos que forman un bucle cerrado que no forma intersección consigo mismo. Un polígono puede contener varios anillos externos. El orden de vértices o la orientación de un anillo indica qué parte del anillo es el interior del polígono. La parte que queda a la derecha del observador que camina por el anillo en orden de vértices es la parte interna del polígono. Los vértices de los anillos que definen orificios en polígonos están en una dirección contraria a la de las agujas del reloj. Los vértices correspondientes a un solo polígono con anillos están, por tanto, siempre en el orden de las agujas del reloj. Los anillos de un polígono se denominan partes.

Puesto que esta especificación no prohíbe puntos consecutivos con coordenadas idénticas, los lectores de archivos de forma deben manejar estos casos. Por otro lado, no se permiten las partes degeneradas de longitud cero o de área cero que pueden derivar de dichos casos.

Los campos de un polígono son:

Box El recuadro delimitador del polígono que se almacena en el orden Xmín, Ymín, Xmáx, Ymáx.

NumParts
El número de anillos del polígono.

NumPoints
El número total de puntos correspondiente a todos los anillos.

Parts Una matriz de longitud NumParts. Almacena, para cada anillo, el índice de su primer punto en la matriz de puntos. Los índices de la matriz son con respecto a 0.

Points Una matriz de longitud NumPoints. Los puntos de cada anillo del polígono se almacenan de extremo a extremo. Los puntos correspondientes al anillo 2 siguen a los correspondientes al anillo 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del punto inicial correspondiente a cada anillo. No hay ningún delimitador en la matriz de puntos entre anillos.

Notas importantes sobre las formas Polygon:

- Los anillos son cerrados (el primer y último vértice de un anillo DEBE ser el mismo).
- El orden de anillos en la matriz de puntos no es significativo.
- Los polígonos almacenados en un archivo de formas deben estar limpios. Un polígono limpio es uno que:
  - No forma intersección consigo mismo. Esto significa que un segmento que pertenece a un anillo no puede formar intersección con un segmento que pertenece a otro anillo. Los anillos de un

polígono se pueden tocar entre sí por los vértices, pero no a lo largo de los segmentos. Se considera que los segmentos colineales forman intersección.

– Tiene el interior del polígono en el lado ″correcto″ de la línea que lo define. La parte que queda a la derecha de un observador que camina por el anillo en orden de vértices es la parte interior del polígono. Los vértices correspondientes a un solo polígono con anillos están, por tanto, siempre en el orden de las agujas del reloj. Los anillos que definen orificios en estos polígonos tienen una orientación contraria a la de las agujas del reloj.

Los polígonos ″sucios″ son aquellos cuyos anillos que definen orificios en el polígono también van en el sentido de las agujas del reloj, lo que causa que los interiores se solapen.

Instancia de polígono de ejemplo:

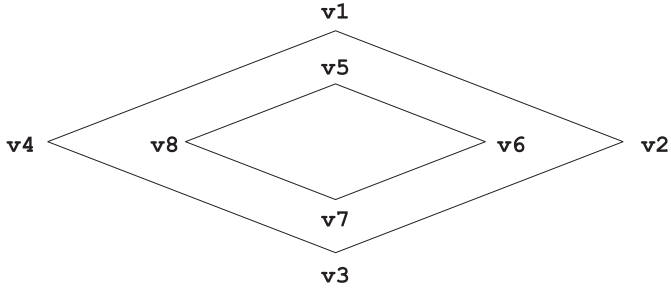


Figura 40. Polígono con un orificio y ocho vértices

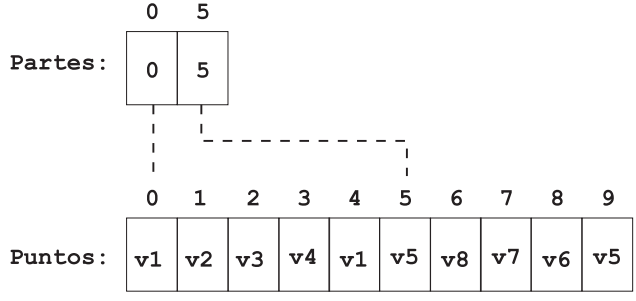


Figura 41. Contenido de la corriente de bytes del polígono. NumParts igual a 2 y NumPoints igual a 10. Observe que el orden de los puntos del anillo interno del polígono está invertido.

Tabla 70. Contenido de la corriente de bytes de un polígono

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	5	Integer	1	Little

Tabla 70. Contenido de la corriente de bytes de un polígono (continuación)

Posición	Campo	Valor	Tipo	Número	Orden
Byte 4	Box	Box	Double	4	Little
Byte 36	NumParts	NumParts	Integer	1	Little
Byte 40	NumPoints	NumPoints	Integer	1	Little
Byte 44	Parts	Parts	Integer	NumParts	Little
Byte X	Points	Points	Point	NumPoints	Little

Nota: X = 44 + 4 * NumParts.

Tipos de formas con medidas en espacio XY

PointM

Un PointM consta de un par de coordenadas de doble precisión en el orden X, Y, más una medida M.

Tabla 71. Contenido de la corriente de bytes de PointM

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	21	Integer	1	Little
Byte 4	X	X	Double	1	Little
Byte 12	Y	Y	Double	1	Little
Byte 20	M	M	Double	1	Little

MultiPointM

Los campos de un MultiPointM son:

Box El recuadro delimitador del MultiPointM almacenado en el orden Xmín, Ymín, Xmáx, Ymáx.

NumPoints
El número de puntos.

Points Una matriz de puntos de longitud NumPoints.

NumMs
El número de medidas que siguen. NumMs sólo puede tener dos valores: cero si este campo no va seguido de ninguna medida, e igual a NumPoints si hay medidas.

M Range
Las medidas mínima y máxima correspondientes a MultiPointM almacenadas en el orden Mmín, Mmáx.

M Array
Una matriz de medidas de longitud NumPoints.

Tabla 72. Contenido de la corriente de bytes de MultiPointM

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	28	Integer	1	Little
Byte 4	Box	Box	Double	4	Little
Byte 36	NumPoints	NumPoints	Integer	1	Little
Byte 40	Points	Points	Point	NumPoints	Little
Byte X	NumMs	NumMs	Integer	1	Little
Byte X+4*	Mmin	Mmin	Double	1	Little
Byte X+12*	Mmax	Mmax	Double	1	Little
Byte X+20*	Marray	Marray	Double	NumPoints	Little

Notas:

1. X = 40 + (16 * NumPoints)
2. * opcional

PolyLineM

Un archivo de formas PolyLineM consta de una o más partes. Una parte es una secuencia conectada de dos o más puntos. Las partes pueden estar o no conectadas entre sí. Las partes pueden formar o no formar intersección entre sí.

Los campos de un PolyLineM son:

Box El recuadro delimitador del PolyLineM almacenado en el orden Xmín, Ymín, Xmáx, Ymáx.

NumParts
El número de partes del PolyLineM.

NumPoints
El número total de puntos correspondiente a todas las partes.

Parts Una matriz de longitud NumParts. Almacena, para cada parte, el índice de su primer punto en la matriz de puntos. Los índices de la matriz son con respecto a 0.

Points Una matriz de longitud NumPoints. Los puntos de cada parte de la geometría PolyLineM se almacenan de extremo a extremo. Los puntos correspondientes a la parte 2 siguen a los correspondientes a la parte 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del punto inicial correspondiente a cada parte. No hay ningún delimitador en la matriz de puntos entre partes.

NumMs

El número de medidas que siguen. NumMs sólo puede tener dos valores: cero si este campo no va seguido de ninguna medida, e igual a NumPoints si hay medidas.

M Range

Las medidas mínima y máxima correspondientes a PolyLineM almacenadas en el orden Mmín, Mmáx.

M Array

Una matriz de longitud NumPoints. Las medidas de cada parte de la geometría PolyLineM se almacenan de extremo a extremo. Las medidas correspondientes a la parte 2 siguen a las correspondientes a la parte 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del punto inicial correspondiente a cada parte. No hay ningún delimitador en la matriz de medidas entre partes.

Tabla 73. Contenido de la corriente de bytes de PolyLineM

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	13	Integer	1	Little
Byte 4	Box	Box	Double	4	Little
Byte 36	NumParts	NumParts	Integer	1	Little
Byte 40	NumPoints	NumPoints	Integer	1	Little
Byte 44	Parts	Parts	Integer	NumParts	Little
Byte X	Points	Points	Point	NumPoints	Little
Byte Y	NumMs	NumMs	Integer	1	Little
Byte Y+4*	Mmin	Mmin	Double	1	Little
Byte Y+12*	Mmax	Mmax	Double	1	Little
Byte Y+20*	Marray	Marray	Double	NumPoints	Little

Notas:

1. X = 44 + (4 * NumParts), Y = X + (16 * NumPoints).
2. * opcional

PolygonM

Un PolygonM consta de un número de anillos. Un anillo es un bucle cerrado que no forma intersección consigo mismo. Observe que las intersecciones se calculan en espacio XY, *no* en espacio XYM. Un PolygonM puede contener varios anillos externos. Los anillos de un PolygonM se denominan partes.

Los campos de un PolygonM son:

Box El recuadro delimitador del PolygonM almacenado en el orden Xmín, Ymín, Xmáx, Ymáx.

NumParts
El número de anillos del PolygonM.

NumPoints
El número total de puntos correspondiente a todos los anillos.

Parts Una matriz de longitud NumParts. Almacena, para cada anillo, el índice de su primer punto en la matriz de puntos. Los índices de la matriz son con respecto a 0.

Points Una matriz de longitud NumPoints. Los puntos de cada anillo del PolygonM se almacenan de extremo a extremo. Los puntos correspondientes al anillo 2 siguen a los correspondientes al anillo 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del punto inicial correspondiente a cada anillo. No hay ningún delimitador en la matriz de puntos entre anillos.

NumMs
El número de medidas que siguen. NumMs sólo puede tener dos valores: cero si este campo no va seguido de ninguna medida o igual a NumPoints si hay medidas.

M Range
Las medidas mínima y máxima correspondientes a PolygonM almacenadas en el orden Mmín, Mmáx.

M Array
Una matriz de longitud NumPoints. Las medidas de cada anillo del PolygonM se almacenan de extremo a extremo. Las medidas correspondientes al anillo 2 siguen a las correspondientes al anillo 1, y así sucesivamente. La matriz de partes alberga el índice de matriz de la medida inicial correspondiente a cada anillo. No hay ningún delimitador en la matriz de medidas entre anillos.

Notas importantes sobre las formas PolygonM:

- Los anillos son cerrados (el primer y el último vértice de un anillo debe ser el mismo).
- El orden de anillos en la matriz de puntos no es significativo.

Tabla 74. Contenido de la corriente de bytes de PolygonM

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	15	Integer	1	Little
Byte 4	Box	Box	Double	4	Little
Byte 36	NumParts	NumParts	Integer	1	Little
Byte 40	NumPoints	NumPoints	Integer	1	Little

Tabla 74. Contenido de la corriente de bytes de PolygonM (continuación)

Posición	Campo	Valor	Tipo	Número	Orden
Byte 44	Parts	Parts	Integer	NumParts	Little
Byte X	Points	Points	Point	NumPoints	Little
Byte Y	NumMs	NumMs	Integer	1	Little
Byte Y+4*	Mmin	Mmin	Double	1	Little
Byte Y+12*	Mmax	Mmax	Double	1	Little
Byte Y+20*	Marray	Marray	Double	NumPoints	Little

Notas:

1. X = 44 + (4 * NumParts), Y = X + (16 * NumPoints).
2. * opcional

Tipos de formas en espacio XYZ

PointZ

Un PointZ consta de tres coordenadas de doble precisión en el orden X, Y, Z más una medida.

Tabla 75. Contenido de la corriente de bytes de PointZ

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	11	Integer	1	Little
Byte 4	X	X	Double	1	Little
Byte 12	Y	Y	Double	1	Little
Byte 20	Z	Z	Double	1	Little
Byte 28	Measure	M	Double	1	Little

MultiPointZ

Un MultiPointZ representa una serie de PointZ, del siguiente modo:

- El recuadro delimitador se almacena en el orden Xmín, Ymín, Xmáx, Ymáx.
- El rango de Z límite se almacena en el orden Zmín, Zmáx. El rango de M límite se almacena en el orden Mmín, Mmáx.

Tabla 76. Contenido de la corriente de bytes de MultiPointZ

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	18	Integer	1	Little
Byte 4	Box	Box	Double	4	Little
Byte 36	NumPoints	NumPoints	Integer	1	Little
Byte 40	Points	Points	Point	NumPoints	Little

Tabla 76. Contenido de la corriente de bytes de MultiPointZ (continuación)

Posición	Campo	Valor	Tipo	Número	Orden
Byte X	Zmin	Zmin	Double	1	Little
Byte X+8	Zmax	Zmax	Double	1	Little
Byte X+16	Zarray	Zarray	Double	NumPoints	Little
Byte Y	NumMs	NumMs	Integer	1	Little
Byte Y+4*	Mmin	Mmin	Double	1	Little
Byte Y+12*	Mmax	Mmax	Double	1	Little
Byte Y+20*	Marray	Marray	Double	NumPoints	Little

Notas:

1. X = 40 + (16 * NumPoints); Y = X + 16 + (8 * NumPoints)
2. * opcional

PolyLineZ

Un PolyLineZ consta de una o más partes. Una parte es una secuencia conectada de dos o más puntos. Las partes pueden estar o no conectadas entre sí. Las partes pueden formar o no formar intersección entre sí.

Los campos de un PolyLineZ son:

Box El recuadro delimitador del PolyLineZ almacenado en el orden Xmín, Ymín, Xmáx, Ymáx.

NumParts
El número de partes del PolyLineZ.

NumPoints
El número total de puntos correspondiente a todas las partes.

Parts Una matriz de longitud NumParts. Almacena, para cada parte, el índice de su primer punto en la matriz de puntos. Los índices de la matriz son con respecto a 0.

Points Una matriz de longitud NumPoints. Los puntos de cada parte de la geometría PolyLineZ se almacenan de extremo a extremo. Los puntos correspondientes a la parte 2 siguen a los correspondientes a la parte 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del punto inicial correspondiente a cada parte. No hay ningún delimitador en la matriz de puntos entre partes.

Z Range
Los valores mínimo y máximo de Z correspondientes a PolyLineZ almacenados en el orden Zmín, Zmáx.

Z Array

Una matriz de longitud NumPoints. Los valores de Z para cada parte del PolyLineZ se almacenan de extremo a extremo. Los valores de Z correspondientes a la parte 2 siguen a los valores de Z correspondientes a la parte 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del punto inicial correspondiente a cada parte. No hay ningún delimitador en la matriz Z entre partes.

NumMs

El número de medidas que siguen. NumMs sólo puede tener dos valores: cero si este campo no va seguido de ninguna medida, e igual a NumPoints si hay medidas.

M Range

Las medidas mínima y máxima correspondientes a PolyLineZ almacenadas en el orden Mmín, Mmáx.

M Array

Una matriz de longitud NumPoints. Las medidas para cada parte del PolyLineZ se almacenan de extremo a extremo. Las medidas correspondientes a la parte 2 siguen a las correspondientes a la parte 1, y así sucesivamente. La matriz de partes alberga el índice de matriz de la medida inicial correspondiente a cada parte. No hay ningún delimitador en la matriz de medidas entre partes.

Tabla 77. Contenido de la corriente de bytes de la PolyLineZ

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	13	Integer	1	Little
Byte 4	Box	Box	Double	4	Little
Byte 36	NumParts	NumParts	Integer	1	Little
Byte 40	NumPoints	NumPoints	Integer	1	Little
Byte 44	Parts	Parts	Integer	NumParts	Little
Byte X	Points	Points	Point	NumPoints	Little
Byte Y	Zmin	Zmin	Double	1	Little
Byte Y+8	Zmax	Zmax	Double	1	Little
Byte Y+16	Zarray	Zarray	Double	NumPoints	Little
Byte Z	NumMs	NumMs	Integer	1	Little
Byte Z+4*	Mmin	Mmin	Double	1	Little
Byte Z+12*	Mmax	Mmax	Double	1	Little
Byte Z+20*	Marray	Marray	Double	NumPoints	Little

Notas:

1. X = 44 + (4 * NumParts), Y = X + (16 * NumPoints), Z = Y + 16 + (8 * NumPoints)
2. * opcional

PolygonZ

Un PolygonZ consta de varios anillos. Un anillo es un bucle cerrado que no forma intersección consigo mismo. Un PolygonZ puede contener varios anillos externos. Los anillos de un PolygonZ se denominan partes.

Los campos de un PolygonZ son:

Box Recuadro delimitador del PolygonZ almacenado en el orden Xmín, Ymín, Xmáx, Ymáx.

NumParts
Número de anillos del PolygonZ.

NumPoints
Número total de puntos correspondientes a todos los anillos.

Parts Matriz de longitud NumParts. Almacena, para cada anillo, el índice de su primer punto en la matriz de puntos. Los índices de la matriz están referidos a 0.

Points Matriz de longitud NumPoints. Los puntos de cada anillo del PolygonZ se almacenan de forma consecutiva. Los puntos correspondientes al anillo 2 siguen a los correspondientes al anillo 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del punto inicial correspondiente a cada anillo. No hay ningún delimitador en la matriz de puntos entre anillos.

Z Range
Los valores mínimo y máximo de Z correspondientes al arco almacenados en el orden Zmín, Zmáx.

Z Array
Matriz de longitud NumPoints. Los valores Z de cada anillo del PolygonZ se almacenan de forma consecutiva. Los valores de Z correspondientes al anillo 2 siguen a los valores de Z correspondientes al anillo 1, y así sucesivamente. La matriz de partes alberga el índice de matriz del valor Z inicial correspondiente a cada anillo. No hay ningún delimitador en la matriz de valores Z entre anillos.

NumMs
Número de medidas que siguen. NumMs sólo puede tener dos valores: cero si este campo no va seguido de ninguna medida, o igual a NumPoints si hay medidas presentes.

M Range

Medidas mínima y máxima correspondientes a PolygonZ almacenadas en el orden Mmín, Mmáx.

M Array

Matriz de longitud NumPoints. Las medidas de cada anillo del PolygonZ se almacenan de forma consecutiva. Las medidas correspondientes al anillo 2 siguen a las correspondientes al anillo 1, y así sucesivamente. La matriz de partes alberga el índice de matriz de la medida inicial correspondiente a cada anillo. No hay ningún delimitador en la matriz de medidas entre anillos.

Notas importantes sobre las formas del PolygonZ:

- Los anillos son cerrados (el primer y último vértice de un anillo DEBE ser el mismo).
- El orden de los anillos en la matriz de puntos no es significativo.

Tabla 78. Contenido de la corriente de bytes de PolygonZ

Posición	Campo	Valor	Tipo	Número	Orden
Byte 0	Shape Type	15	Integer	1	Little
Byte 4	Box	Box	Double	4	Little
Byte 36	NumParts	NumParts	Integer	1	Little
Byte 40	NumPoints	NumPoints	Integer	1	Little
Byte 44	Parts	Parts	Integer	NumParts	Little
Byte X	Points	Points	Point	NumPoints	Little
Byte Y	Zmin	Zmin	Double	1	Little
Byte Y+8	Zmax	Zmax	Double	1	Little
Byte Y+16	Zarray	Zarray	Double	NumPoints	Little
Byte Z	NumMs	NumMs	Integer	1	Little
Byte Z+4*	Mmin	Mmin	Double	1	Little
Byte Z+12*	Mmax	Mmax	Double	1	Little
Byte Z+20*	Marray	Marray	Double	NumPoints	Little

Parte 3. Apéndices

Avisos

Es posible que IBM no comercialice en todos los países algunos productos, servicios o características descritos en este manual. Consulte al representante local de IBM para obtener información sobre los productos y servicios que actualmente pueden adquirirse en su zona geográfica. Cualquier referencia a un producto, programa o servicio de IBM no pretende afirmar ni implicar que sólo se puede utilizar dicho producto, programa o servicio de IBM. En su lugar se puede utilizar cualquier producto, programa o servicio funcionalmente equivalente que no infrinja ninguno de los derechos de propiedad intelectual de IBM. Sin embargo, es responsabilidad del usuario evaluar y verificar el funcionamiento de cualquier producto, programa o servicio que no sea de IBM.

IBM puede tener patentes o solicitudes de patentes en tramitación que afecten al tema tratado en este documento. La posesión de este documento no confiere ninguna licencia sobre dichas patentes. Puede realizar consultas sobre licencias escribiendo a:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
EE.UU.

En el caso de consultas sobre licencias referentes a información de doble byte (DBCS), consulte al Departamento de Propiedad Intelectual de IBM en su país o envíe consultas por escrito a:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japón

El párrafo siguiente no es aplicable al Reino Unido ni a ningún país en el que tales disposiciones sean incompatibles con la legislación local:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL", SIN GARANTÍA DE NINGUNA CLASE, NI EXPLÍCITA NI IMPLÍCITA, INCLUIDAS, PERO SIN LIMITARSE A ELLAS, LAS GARANTÍAS IMPLÍCITAS DE NO VULNERACIÓN DE DERECHOS, COMERCIABILIDAD O IDONEIDAD PARA UN FIN DETERMINADO. Algunos estados no permiten la exclusión de garantías expresas o implícitas en determinadas transacciones, por lo que es posible que esta declaración no sea aplicable en su caso.

Esta publicación puede contener inexactitudes técnicas o errores tipográficos. Periódicamente se efectúan cambios en la información aquí contenida; dichos cambios se incorporarán a las nuevas ediciones de la publicación. IBM puede efectuar, en cualquier momento y sin previo aviso, mejoras y/o cambios en los productos y/o programas descritos en esta publicación.

Las referencias hechas en esta publicación a sitios Web que no son de IBM se proporcionan sólo para la comodidad del usuario y no constituyen un aval de esos sitios Web. La información contenida en esos sitios Web no forma parte de la información del presente producto IBM y el usuario es responsable de la utilización de esos sitios Web.

IBM puede utilizar o distribuir cualquier información que se le facilite de la manera que considere adecuada, sin contraer por ello ninguna obligación con el remitente.

Los licenciatarios de este programa que deseen obtener información sobre él con el fin de habilitar: (i) el intercambio de información entre programas creados de forma independiente y otros programas (incluido este) y (ii) el uso mutuo de la información intercambiada, deben ponerse en contacto con:

IBM Canada Limited
Office of the Lab Director
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
CANADÁ

Dicha información puede estar disponible, sujeta a los términos y condiciones apropiados, incluido en algunos casos, el pago de una tarifa.

El programa bajo licencia descrito en este manual y todo el material bajo licencia asociado a él, los proporciona IBM según los términos del Acuerdo de Cliente de IBM, el Acuerdo Internacional de Programas Bajo Licencia de IBM o cualquier acuerdo equivalente entre el usuario e IBM.

Los datos de rendimiento contenidos en este documento se obtuvieron en un entorno controlado. Por tanto, los resultados obtenidos en otros entornos operativos pueden variar significativamente. Algunas mediciones pueden haberse hecho en sistemas experimentales y no es seguro que estas mediciones sean las mismas en los sistemas disponibles comercialmente. Además, algunas mediciones pueden haberse calculado mediante extrapolación. Los resultados reales pueden variar. Los usuarios del presente manual deben verificar los datos aplicables para su entorno específico.

La información referente a productos que no son de IBM se ha obtenido de los proveedores de esos productos, de sus anuncios publicados o de otras

fuentes disponibles públicamente. IBM no ha probado esos productos y no puede confirmar la exactitud del rendimiento, la compatibilidad ni cualquier otra afirmación referente a productos que no son de IBM. Las preguntas sobre las prestaciones de productos que no son de IBM deben dirigirse a los proveedores de esos productos.

Todas las declaraciones de intenciones de IBM están sujetas a cambio o cancelación sin previo aviso, y sólo representan objetivos.

Esta publicación puede contener ejemplos de datos e informes que se utilizan en operaciones comerciales diarias. Para ilustrarlos de la forma más completa posible, los ejemplos incluyen nombres de personas, empresas, marcas y productos. Todos estos nombres son ficticios y cualquier similitud con nombres y direcciones utilizados por una empresa real es totalmente fortuita.

LICENCIA DE COPYRIGHT:

Este manual puede contener programas de aplicaciones de ejemplo escritos en lenguaje fuente, que muestran técnicas de programación en diversas plataformas operativas. Puede copiar, modificar y distribuir estos programas de ejemplo de la forma que desee, sin pago alguno a IBM, con la intención de desarrollar, utilizar, comercializar o distribuir programas de aplicaciones de acuerdo con la interfaz de programación de aplicaciones correspondiente a la plataforma operativa para la que están escritos los programas de ejemplo. Estos ejemplos no se han probado exhaustivamente bajo todas las condiciones. Por tanto, IBM no puede asegurar ni implicar la fiabilidad, utilidad o función de estos programas.

Cada copia o porción de estos programas de ejemplo o cualquier trabajo derivado debe incluir una nota de copyright como la siguiente:

© (nombre de la empresa) (año). Partes de este código derivan de programas de ejemplo de IBM Corp. © Copyright IBM Corp. _especifique el año o años _. Reservados todos los derechos.

Marcas registradas

Los términos siguientes, que pueden estar indicados por un asterisco (*), son marcas registradas de International Business Machines Corporation en los Estados Unidos y/o en otros países.

ACF/VTAM
AISPO
AIX
AIX/6000
AIXwindows
AnyNet
APPN
AS/400
BookManager
CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eNetwork
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
LAN DistanceMVS
MVS/ESA
MVS/XA
Net.Data
OS/2
OS/390
OS/400
PowerPC
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/DS
SQL/400
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WIN-OS/2

Los términos siguientes son marcas registradas de otras empresas:

Microsoft, Windows y Windows NT son marcas registradas de Microsoft Corporation.

Java, y las marcas registradas y logotipos basados en Java y Solaris, son marcas registradas de Sun Microsystems, Inc. en los Estados Unidos y/o en otros países.

Tivoli y NetView son marcas registradas de Tivoli Systems Inc. en los Estados Unidos y/o en otros países.

UNIX es una marca registrada en los Estados Unidos y/o en otros países bajo licencia exclusiva de X/Open Company Limited.

Otros nombres de empresas, productos o servicios, que pueden estar indicados por un doble asterisco (**), pueden ser marcas registradas o marcas de servicio de otras empresas.

Índice

E

F

G

H

I

J

L

M

P

R

S

T

U

V

W

X

Y

Z

Cómo ponerse en contacto con IBM

Si tiene un problema técnico, repase y lleve a cabo las acciones que se sugieren en la *Guía de resolución de problemas* antes de ponerse en contacto con el Centro de Asistencia al Cliente de DB2. Dicha guía sugiere información que puede reunir para ayudar al Centro de Asistencia a proporcionarle un mejor servicio.

Para obtener información o para solicitar cualquiera de los productos de DB2 Universal Database, consulte a un representante de IBM de una sucursal local o a un concesionario autorizado de IBM.

Si vive en los Estados Unidos, puede llamar a uno de los números siguientes:

- 1-800-237-5511 para obtener soporte técnico
- 1-888-426-4343 para obtener información sobre las opciones de servicio técnico disponibles

Información sobre productos

Si vive en los Estados Unidos, puede llamar a uno de los números siguientes:

- 1-800-IBM-CALL (1-800-426-2255) o 1-800-3IBM-OS2 (1-800-342-6672) para solicitar productos u obtener información general.
- 1-800-879-2755 para solicitar publicaciones.

http://www.ibm.com/software/data/
Las páginas Web de DB2 ofrecen información actual sobre DB2 referente a novedades, descripciones de productos, planes de formación, etc.

http://www.ibm.com/software/data/db2/library/
La biblioteca técnica de servicio y de productos DB2 ofrece acceso a las preguntas más frecuentes (FAQ), arreglos de programa, manuales e información técnica actualizada sobre DB2.

Nota: Puede que esta información sólo esté disponible en inglés.

http://www.elink.ibmlink.ibm.com/pbl/pbl/
El sitio Web para el pedido de publicaciones internacionales proporciona información sobre cómo hacer pedidos de manuales.

http://www.ibm.com/education/certify/
El Programa de homologación profesional contenido en el sitio Web de IBM proporciona información de prueba de homologación para diversos productos de IBM, incluido DB2.

ftp.software.ibm.com
Inicie la sesión como anónimo (anonymous). En el directorio /ps/products/db2 encontrará programas de demostración, arreglos de programa, información y herramientas referentes a DB2 y a muchos otros productos.

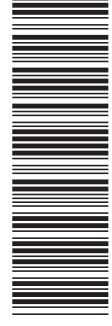
comp.databases.ibm-db2, bit.listserv.db2-l
En estos foros de discusión de Internet los usuarios pueden explicar sus experiencias con los productos DB2.

En Compuserve: GO IBMDB2
Entre este mandato para acceder a los foros referentes a la familia de productos DB2. Todos los productos DB2 tienen soporte a través de estos foros.

Para conocer cómo ponerse en contacto con IBM desde fuera de los Estados Unidos, consulte el Apéndice A del manual *IBM Software Support Handbook*. Para acceder a este documento, vaya a la página Web siguiente: http://www.ibm.com/support/ y luego seleccione el enlace "IBM Software Support Handbook", cerca del final de la página.

Nota: En algunos países, los distribuidores autorizados de IBM deben ponerse en contacto con su organización de soporte en lugar de acudir al Centro de Asistencia de IBM.

Número Pieza: CT7M2ES



SC10-3528-01

Spine information:

IBM® DB2® Spatial Extender

DB2 Spatial Extender Guía y consulta del usuario

Versión 7