IBM® DB2® Spatial Extender e
Geodetic Extender

Referência e Manual do Usuário

Versão 8.2

S517-7368-01

IBM® DB2® Spatial Extender e
Geodetic Extender

Referência e Manual do Usuário

Versão 8.2

S517-7368-01

Antes de utilizar estas informações e o produto a suportado por elas, certifique-se de ter lido as informações gerais em *Avisos*.

Você pode solicitar publicações da IBM on-line ou através do representante IBM local.

- Para solicitar publicações on-line, acesse o IBM Publications Center em www.ibm.com/shop/publications/order
- Para localizar o representante IBM local, acesse o IBM Directory of Worldwide Contacts em www.ibm.com/planetwide

Para solicitar publicações do DB2 através do Departamento de Marketing e Vendas nos Estados Unidos e Canadá, ligue para 1-800-IBM-4YOU (426-4968). No Brasil, ligue para 0-800-7014-262.

Quando o Cliente envia seus comentários, concede direitos, não exclusivos, à IBM para usá-los ou distribuí-los da maneira que achar conveniente, sem que isso implique em qualquer compromisso ou obrigação para com o Cliente.

Índice

Índice

Parte 1. Introdução

Capítulo 1. Sobre o DB2 Spatial Extender

Este capítulo apresenta o DB2 Spatial Extender explicando sua finalidade, descrevendo os dados que ele suporta e explicando como seus conceitos básicos se adaptam.

A finalidade do DB2 Spatial Extender

Utilize o DB2® Spatial Extender para gerar e analisar informações espaciais sobre recursos geográficos e para armazenar e gerenciar os dados nos quais estas informações são baseadas. Um *recurso geográfico* (às vezes chamado de *recurso* nesta discussão, para abreviar) é algo no mundo real que tem uma localização identificável, ou algo que pode ser imaginado como existente em uma localização identificável. Um recurso pode ser:

- Um objeto (ou seja, uma entidade concreta de qualquer tipo); por exemplo, um rio, uma floresta ou uma cadeia de montanhas.
- Um espaço, uma zona de segurança em torno de um local perigoso, ou uma área de marketing atendida por um determinado ramo de negócios.
- Um evento que ocorre em uma localização que pode ser definida; por exemplo, um acidente automobilístico que ocorreu em um determinado cruzamento, ou uma transação de vendas em uma loja específica.

Existem recursos em vários ambientes. Por exemplo, os objetos mencionados na lista anterior — rio, floresta, cadeia de montanhas — pertencem ao ambiente natural. Outros objetos, como cidades, edifícios e escritórios pertencem ao ambiente cultural. Ainda existem outros, como parques, zoológicos e zonas rurais que representam uma combinação dos ambientes natural e cultural.

Nesta discussão, o termo *informações espaciais* refere-se ao tipo de informação que o DB2 Spatial Extender disponibiliza para os seus usuários — ou seja, fatos e figuras sobre as localizações de recursos geográficos. Exemplos de informações espaciais são:

- Localizações de recursos geográficos no mapa (por exemplo, valores longitudinais e latitudinais que definem onde as cidades estão situadas)
- A localização de recursos geográficos em relação um ao outro (por exemplo, pontos dentro de uma cidade onde hospitais e clínicas estão localizados, ou a proximidade das residências da cidade em relação a zonas de terremoto)
- Modos como os recursos geográficos estão relacionados entre si (por exemplo, informações de que um determinado sistema fluvial está contido dentro de um região específica, ou de que determinadas pontes naquela região atravessam os braços do sistema fluvial)
- Medidas que se aplicam a um ou mais recursos geográficos (por exemplo, a distância entre um prédio de escritórios e sua divisão de terreno ou o comprimento de um perímetro de preservação de um pássaro)

Informações espaciais, isoladas ou em conjunto com dados relacionais tradicionais, podem ajudá-lo nas atividades como definir as áreas nas quais você oferece serviços e determinar localizações de possíveis mercados. Suponha, por exemplo, que o gerente de um distrito municipal precise verificar quais requerentes e

beneficiários realmente moram dentro da área atendida pelo distrito. O DB2 Spatial Extender pode derivar estas informações da localização da área atendida e dos endereços de requerentes e beneficiários.

Ou suponha que o proprietário de uma cadeia de restaurantes queira fazer negócio em cidades próximas. Para determinar onde abrir novos restaurantes, o proprietário precisa responder a perguntas como: Em que locais destas cidades está concentrada a clientela que geralmente freqüenta meus restaurantes? Onde ficam as principais estradas? Onde é mais baixa a taxa de crimes? Onde se encontram os restaurantes da concorrência? O DB2 Spatial Extender e o DB2 podem gerar informações para responder estas perguntas. Além disso, ferramentas front-end, embora não necessárias, podem participar. Para ilustrar: uma ferramenta de visualização pode colocar informações geradas pelo DB2 Spatial Extender — por exemplo, a localização de concentrações de clientes e a proximidade de rodovias principais a bons restaurantes — no formato de um gráfico em um mapa. As ferramentas de inteligência de negócios podem colocar informações associadas — por exemplo, nomes e descrições de restaurantes concorrentes — em formato de relatório.

Dados Espaciais e Geodésicos

Esta seção apresenta uma visão geral dos dados que são gerados, armazenados e manipulados para obtenção de informações espaciais. Os tópicos abrangidos são:

- Como os dados representam os recursos geográficos
- A natureza de dados espaciais e de dados geodésicos
- Formas de produção de dados espaciais

Como os Dados Representam os Recursos Geográficos

No DB2® Spatial Extender, um recurso geográfico pode ser representado por um ou mais itens de dados; por exemplo, os itens de dados em uma linha de uma tabela. (Um *item de dados* é o valor ou valores que ocupam uma célula de uma tabela relacional.) Por exemplo, considere edifícios comerciais e residências. Na Figura 1, cada linha da tabela FILIAIS representa uma filial de um banco. De forma semelhante, cada linha da tabela CLIENTES na Figura 1, tomada por inteiro, representa um cliente do banco. No entanto, um subconjunto de cada linha — especificamente os itens de dados que constituem o endereço de um cliente — representam a residência do cliente.

FILIAIS

ID	NOME	ENDEREÇO	CIDADE	CEP	ESTADO	PAÍS
937	Airzone-Multern	92467 Airzone Blvd	San Jose	95141	CA	USA

CLIENTES

ID	SOBRE-NOME	NOME	ENDEREÇO	CIDADE	CEP	ESTADO	PAÍS	CONTA CORRENTE	POUPANÇA
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	95141	CA	USA	A	A

Figura 1. Dados que Representam Recursos Geográficos. A linha de dados na tabela FILIAIS representa uma filial de um banco. Os dados do endereço na tabela CLIENTES representam a residência de um cliente. Os nomes e endereços em ambas as tabelas são fictícios.

As tabelas na Figura 1 na página 4 contêm dados que identificam e descrevem as filiais e clientes do banco. Esta discussão refere-se a estes dados como *dados de negócios.*

Um subconjunto de dados de negócios — os valores que indicam os endereços de filiais e de clientes — pode ser convertido em valores a partir dos quais as informações espaciais são geradas. Por exemplo, como mostrado em Figura 1 na página 4, o endereço de uma das filiais é 92467 Airzone Blvd., San Jose, CA 95141, USA. O endereço de um cliente é 9 Concourt Circle, San Jose, CA 95141, USA. O DB2 Spatial Extender pode converter estes endereços em valores que indicam onde a filial e a residência do cliente estão localizadas com relação uma à outra. A Figura 2 mostra as tabelas FILIAIS e CLIENTES com novas colunas que são projetadas para conter tais valores.

FILIAIS

ID	NOME	ENDEREÇO	CIDADE	CEP	ESTADO	PAÍS	LOCALIZAÇÃO
937	Airzone-Multern	92467 Airzone Blvd	San Jose	95141	CA	USA	

CLIENTES

ID	SOBRE-NOME	NOME	ENDEREÇO	CIDADE	CEP	ESTADO	PAÍS	LOCALIZ.	CONTA CORRENTE	POUPANÇA
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	95141	CA	USA		A	A

Figura 2. Tabelas com Colunas Espaciais Incluídas. Em cada tabela, a coluna LOCALIZAÇÃO irá conter as coordenadas que correspondem aos endereços.

Como as informações espaciais serão derivadas dos itens de dados armazenados na coluna LOCALIZAÇÃO, esses itens de dados são referidos nesta discussão como *dados espaciais.*

A Natureza dos Dados Espaciais

Os dados espaciais são compostos por coordenadas. Uma *coordenada* é um número que indica:

- Uma posição em um eixo relativa a uma origem, especificada uma unidade de comprimento.
- Uma direção relativa a uma linha de base ou plano, especificada uma unidade de medida angular.

Por exemplo, a latitude é uma coordenada que indica um ângulo relativo ao plano equatorial, geralmente em graus. A longitude é uma coordenada que indica um ângulo relativo ao meridiano de Greenwich, geralmente também em graus. Assim, em um mapa, a posição do Parque Nacional Yellowstone é definida por 44,45 graus de latitude ao norte do equador e por 110,40 graus de longitude a oeste do meridiano de Greenwich. Mais precisamente, estas coordenadas se referem ao centro do Parque Nacional Yellowstone nos E.U.A.

As definições de latitude e longitude, seus pontos, linhas e planos de referência, unidades de medida e outros parâmetros associados são referidos coletivamente como um *sistema de coordenadas*. Os sistemas de coordenadas podem ser baseados em valores diferentes da latitude e longitude. Estes sistemas de coordenadas possuem seus próprios pontos, linhas e planos de referência, unidades de medida e parâmetros adicionais associados (como a transformação da projeção). Para obter informações adicionais, consulte “Sistemas de Coordenadas” na página 59.

O item de dados espaciais mais simples consiste em um único par de coordenadas que define a posição de uma única localização geográfica. Um item de dados espaciais mais amplo consiste em várias coordenadas que definem um caminho linear que uma rua ou rio pode formar. Um terceiro tipo consiste em coordenadas que definem o limite de uma área; por exemplo, o limite de um pedaço de terra ou planície aluvial.

Cada item de dados espaciais é uma instância de um tipo de dados espacial. O tipo de dados para coordenadas que marcam uma única localização é ST_Point; o tipo de dados para coordenadas que definem um caminho linear é ST_LineString; e o tipo de dados para coordenadas que definem o limite de uma área é ST_Polygon. Estes tipos, junto com os outros tipos de dados espaciais, são tipos estruturados que pertencem a uma única hierarquia.

A Natureza de Dados Geodésicos

O DB2 Geodetic Extender utiliza os mesmos tipos de dados e funções que o Spatial Extender para armazenar dados geográficos em um banco de dados DB2. Os dados geodésicos são dados espaciais que são expressos em coordenadas de latitude e longitude, em um sistema de coordenadas (consulte “Sistemas de Coordenadas” na página 59) que descreve uma superfície redonda, contínua e fechada. Diferente do Spatial Extender, que trata a Terra como um mapa plano, o Geodetic Extender trata a Terra como um globo que não possui bordas ou linhas nos pólos ou no meridiano de data. Um mapa plano requer coordenadas projetadas para transformar coordenadas esféricas em coordenadas planares. Enquanto isso, o Geodetic Extender utiliza a latitude e longitude em um modelo elipsoidal da superfície da Terra. Cálculos tais como, interseção de linhas, sobreposição de área, distância e área são exatos e precisos, independentemente da localização.

Para obter informações adicionais, consulte “Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender” na página 160 e “Diferenças em Trabalhar com Representações Planas e Redondas da Terra” na página 199.

De Onde Vêm os Dados Espaciais

Os dados espaciais podem ser:

- Derivados de dados de negócios
- Gerados a partir de funções espaciais
- Importados a partir de origens externas

Utilizando Dados de Negócios Como Dados de Origem

O DB2 Spatial Extender pode derivar dados espaciais de dados de negócios, como endereços (conforme mencionado em “Como os Dados Representam os Recursos Geográficos” na página 4). Esse processo é chamado *geocoding*. Para ver a seqüência envolvida, considere a Figura 2 na página 5 como uma imagem “antes” e a Figura 3 na página 7 como uma imagem “depois”. A Figura 2 na página 5 mostra que a tabela FILIAIS e a tabela CLIENTES têm uma coluna designada para dados espaciais. Suponha que o DB2 Spatial Extender efetue geocode nos endereços destas para obter coordenadas que correspondam aos endereços e coloque as coordenadas nas colunas. A Figura 3 na página 7 ilustra este resultado.

FILIAIS

ID	NOME	ENDEREÇO	CIDADE	CEP	ESTADO	PAÍS	LOCALIZAÇÃO
937	Airzone-Multern	92467 Airzone Blvd	San Jose	95141	CA	USA	1653 3094

CLIENTES

ID	SOBRE-NOME	NOME	ENDEREÇO	CIDADE	CEP	ESTADO	PAÍS	LOCALIZ	CONTA CORRENTE	POUPANÇA
59-6396	Kriner	Endela	9 Concourt Circle	San Jose	95141	CA	USA	953 1527	A	A

Figura 3. Tabelas que Contêm Dados Espaciais Derivados de Dados de Origem. A coluna LOCALIZAÇÃO na tabela CLIENTES contém coordenadas derivadas do endereço nas colunas ENDEREÇO, CIDADE, CEP, ESTADO e PAÍS. De forma semelhante, a coluna LOCALIZAÇÃO na tabela FILIAIS contém coordenadas derivadas do endereço nas colunas ENDEREÇO, CIDADE, CEP, ESTADO e PAÍS desta tabela.

O DB2 Spatial Extender utiliza uma função, chamada de *geocoder*, para converter dados de negócios em coordenadas para permitir a operação de funções espaciais nos dados.

Utilizando Funções para Gerar Dados Espaciais

Os dados espaciais podem ser gerados não somente por geocoders, mas também por outras funções. Algumas destas funções, referidas como *construtores*, podem gerar dados espaciais a partir de valores fornecidos como entrada. Outras requerem dados espaciais existentes como entrada. Suponha, por exemplo, que o banco, cujas filiais estão definidas na tabela FILIAIS, deseja saber quantos clientes estão localizados dentro de cinco milhas de cada filial. Antes que o banco obtenha estas informações do banco de dados, ele precisa definir a região que se encontra em um raio especificado ao redor de cada filial. Uma função do DB2 Spatial Extender, ST_Buffer, pode criar esta definição. Utilizando as coordenadas de cada filial como entrada, o ST_Buffer pode gerar as coordenadas que demarcam os perímetros das regiões. A Figura 4 mostra a tabela FILIAIS com informações fornecidas pelo ST_Buffer.

FILIAIS

ID	NOME	ENDEREÇO	CIDADE	CEP	ESTADO	PAÍS	LOCALIZ	ÁREA DE VENDAS
937	Airzone-Multern	92467 Airzone Blvd	San Jose	95141	CA	USA	1653 3094	1002 2001, 1192 3564, 2502 3415, 1915 3394, 1002 2001

Figura 4. Tabela que Contém Novos Dados Espaciais Derivados de Dados Espaciais Existentes. As coordenadas na coluna ÁREA DE VENDAS foram obtidas pela função ST_Buffer a partir das coordenadas na coluna LOCALIZAÇÃO. Assim como as coordenadas na coluna LOCALIZAÇÃO, as na coluna ÁREA DE VENDAS são simuladas; elas não são verdadeiras.

Além do ST_Buffer, o DB2 Spatial Extender fornece várias outras funções que derivam novos dados espaciais de dados espaciais existentes. Para informações adicionais sobre essas funções, consulte “Conceitos relacionados” e “Referência relacionada”, no fim deste tópico.

Importando Dados Espaciais

Um terceiro modo de se obter dados espaciais é importá-los de arquivos fornecidos por origens de dados externas. Estes arquivos geralmente contêm dados que são empregados em mapas: cruzamentos de ruas, planícies aluviais, deslocamentos por terremotos e outros. Utilizando esses dados junto com dados espaciais gerados por você, é possível aumentar as informações espaciais disponíveis. Se, por exemplo,

uma departamento de trabalho público precisa determinar a quais riscos uma comunidade residencial está vulnerável, ele pode usar o ST_Buffer para definir uma região ao redor da comunidade. O departamento de serviço público pode, então, importar dados sobre planícies aluviais e deslocamentos por terremotos para saber quais destas áreas problemáticas abrangem esta região.

Conceitos Relacionados:

- "DB2 Geodetic Extender" na página 159
- "Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender" na página 160
- "Funções Espaciais" na página 299
- "Sistemas de Coordenadas" na página 59
- "Funções que Criam Novas Geometrias com Configurações de Espaço Diferentes" na página 330

Referência Relacionada:

- "Diferenças em Trabalhar com Representações Planas e Redondas da Terra" na página 199
- "Funções Espaciais Suportadas pelo DB2 Geodetic Extender" na página 208
- "ST_Buffer" na página 357
- "Funções que Geram Novas Geometrias de Geometrias Existentes" na página 329

Como Recursos, Informações Espaciais, Dados Espaciais e Geometrias Se Integram

Esta seção resume os diversos conceitos básicos que suportam as operações do DB2® Spatial Extender: recursos geográficos, informações espaciais, dados espaciais e geometrias.

O DB2 Spatial Extender permite obter fatos e figuras relacionados a coisas que podem ser definidas geograficamente — ou seja, em termos de sua localização na terra, ou em uma região da terra. A documentação do DB2 refere-se a tais fatos e figuras como *informações espaciais* e a coisas como *recursos geográficos* (chamados de *recursos* aqui, para abreviar).

Por exemplo, você pode utilizar o DB2 Spatial Extender para determinar se quaisquer áreas ocupadas sobrepõem o local proposto para um aterro. As áreas ocupadas e o local proposto são recursos. Um achado, como por exemplo, se existe alguma sobreposição seria um exemplo de informações espaciais. Se for comprovado que existe a sobreposição, a extensão dela também seria um exemplo de informações espaciais.

Para gerar informações espaciais, o DB2 Spatial Extender deve processar dados que definem as localizações dos recursos. Esses dados, denominados *dados espaciais*, consistem em coordenadas que fazem referência às localizações em um mapa ou projeção semelhante. Por exemplo, para determinar se um recurso sobrepõe outro, o DB2 Spatial Extender deve determinar onde as coordenadas de um dos recursos estão situadas com relação às coordenadas do outro.

No mundo da tecnologia da informação espacial, é comum imaginar recursos como sendo representados por símbolos chamados de *geometrias*. As geometrias são parcialmente visuais e parcialmente matemáticas. Considere seu aspecto visual. O

símbolo para um recurso que tem largura e extensão, como um parque ou cidade, é uma figura com vários lados. Essa geometria é chamada de *polygon*. O símbolo para um recurso linear, como um rio ou estrada, é uma linha. Essa geometria é chamada de *linestring*.

Uma geometria tem propriedades que correspondem a fatos sobre o recurso que ela representa. A maioria destas propriedades podem ser expressas matematicamente. Por exemplo, as coordenadas para um recurso constituem coletivamente uma das propriedades da geometria correspondente do recurso. Outra propriedade, chamada *dimensão*, é um valor numérico que indica se um recurso tem comprimento ou extensão.

Dados espaciais e algumas informações espaciais podem ser exibidos em termos de geometrias. Considere o exemplo, descrito anteriormente, das áreas ocupadas e do local proposto para aterro. Os dados espaciais para as áreas ocupadas incluem coordenadas armazenadas em uma coluna de uma tabela em um banco de dados DB2. A convenção deve considerar o que está armazenado não apenas como dados, mas como geometrias reais. Como as áreas ocupadas têm largura e extensão, você pode ver se estas geometrias são polígonos.

Como dados espaciais, algumas informações espaciais também são exibidas em termos de geometrias. Por exemplo, para determinar se uma área ocupada sobrepõe um local proposto para aterro, o DB2 Spatial Extender deve comparar as coordenadas no polígono que representa o local com as coordenadas dos polígonos que representam as áreas ocupadas. As informações resultantes, isto é, as áreas sobrepostas, também são consideradas polígonos: geometrias com coordenadas, dimensões e outras propriedades.

Capítulo 2. Sobre Geometrias

Este capítulo discute entidades de informações, chamadas geometrias, que consistem em coordenadas e representam recursos geográficos. Os tópicos abrangidos são:

- Geometrias
- Propriedades Geométricas

Geometrias

O Webster's Revised Unabridged Dictionary define *geometria* como "A área da matemática que investiga as relações, propriedades e medidas de figuras sólidas, superfícies, linhas e ângulos; a ciência que trata das propriedades e relações de grandezas; a ciência das relações de espaço." A palavra *geometria* também é utilizada para indicar as recursos geométricos que, no último milênio ou mais, os cartógrafos vêm utilizando para mapear o mundo. Uma definição abstrata desse novo significado de geometria é "um ponto ou agregado de pontos que representam um recurso no solo".

No DB2® Spatial Extender, a definição *operacional* de geometria é "um modelo de recurso geográfico". O modelo pode ser expresso em termos das coordenadas do recurso. O modelo contém informações, por exemplo, as coordenadas identificam a posição do recurso em relação a pontos de referência fixos. Além disso, o modelo pode ser utilizado para produzir informações, por exemplo, a função ST_Overlaps pode utilizar as coordenadas de duas regiões próximas como entrada e retornar informações indicando se as regiões estão sobrepostas ou não.

As coordenadas de um recurso representado pela geometria são consideradas *propriedades* da geometria. Diversos tipos de geometrias têm outras propriedades também, por exemplo, área, comprimento e limites.

As geometrias às quais o DB2 Spatial Extender oferece suporte formam uma hierarquia, mostrada na figura a seguir. A hierarquia da geometria é definida pelo documento OGC (OpenGIS Consortium, Inc.) "OpenGIS Simple Features Specification for SQL". Sete membros da hierarquia são instanciáveis. Ou seja, eles podem ser definidos com valores de coordenadas específicos e processados visualmente conforme mostra a figura.

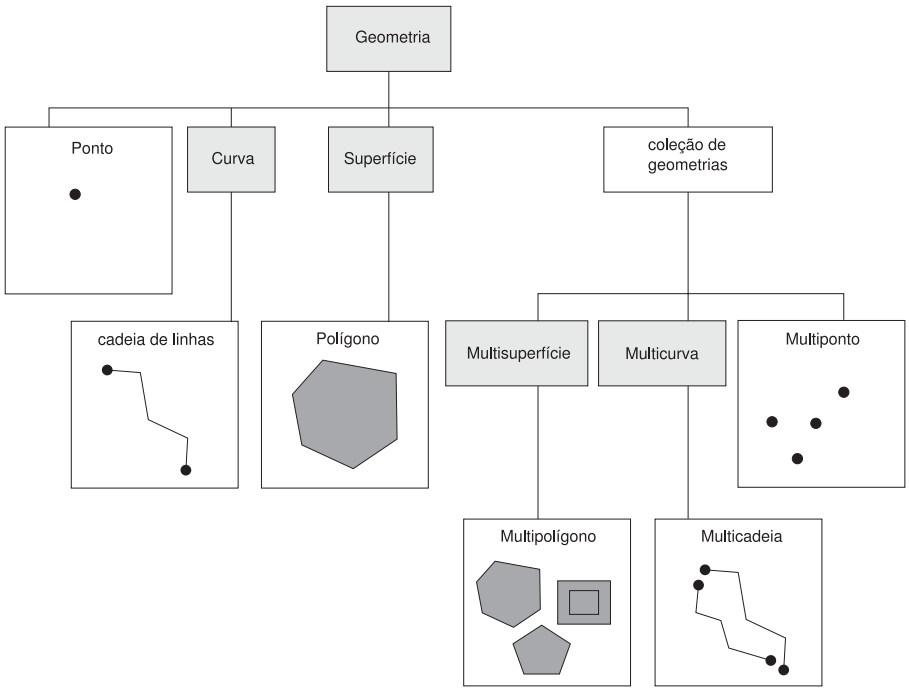


Figura 5. Hierarquia de Geometrias Suportadas pelo DB2 Spatial Extender. As geometrias instanciáveis nesta figura incluem exemplos de como podem ser processadas visualmente.

Os tipos de dados espaciais suportados pelo DB2 Spatial Extender são implementações das geometrias mostradas na figura.

Como a figura indica, uma superclasse chamada *geometria* é a raiz da hierarquia. O tipo raiz e outros subtipos adequados na hierarquia não são instanciáveis. Além disso, os usuários podem definir seus próprios subtipos adequados, instanciáveis ou não.

Os subtipos são divididos em duas categorias: os subtipos de figura geométrica base e os subtipos de coleção homogênea.

As figuras geométricas base incluem:

Pontos
Um único ponto. Os pontos representam recursos distintos que são notados como ocupantes do local em que uma linha da coordenada leste/oeste (como um paralelo) faz interseção com uma linha da coordenada norte/sul (como um meridiano). Por exemplo, suponha que a notação em um mapa-mundi mostra que cada cidade do mapa está localizada na interseção entre um paralelo e um meridiano. Um ponto pode representar cada cidade.

Cadeias de linhas
Uma linha entre dois pontos. Não precisa ser uma linha reta. As cadeias de linhas representam recursos geográficos lineares (por exemplo, ruas, canais e tubulações).

Polígonos
Um polígono ou superfície em um polígono. Polígonos representam recursos geográficos multifacetados (por exemplo, áreas de preservação, florestas e habitats naturais).

As coleções homogêneas incluem:

Multipontos
Uma coleção de geometrias de vários pontos. Multipontos representam recursos de várias partes cujos componentes estão localizados cada um na interseção de uma linha coordenada leste/oeste e uma linha coordenada norte/sul (por exemplo, um arquipélago cujos membros estejam situados cada um em uma interseção de um paralelo e meridiano).

Cadeias multilinha
Uma coleção de geometrias de várias curvas com várias cadeias. Cadeias multilinha representam recursos de várias partes que são compostos (por exemplo, sistemas fluviais e sistemas rodoviários).

Multipolígonos
Uma coleção de geometrias de várias superfícies com vários polígonos. Multipolígonos representam recursos de várias partes compostos de unidades multifacetadas ou componentes (por exemplo, um grupo de fazendas em uma região específica ou um sistema lacustre).

Como seu nome implica, as coleções homogêneas são coleções de figuras geométricas básicas. Além de compartilhar as propriedades da figura geométrica básica, as coleções homogêneas possuem algumas propriedades próprias também.

Conceitos Relacionados:

• "Dados Espaciais e Geodésicos" na página 4

Propriedades Geométricas

Este tópico descreve propriedades geométricas. Estas propriedades são:

- O tipo ao qual uma geometria pertence
- Coordenadas geométricas
- Um interior, limite e exterior da figura geométrica
- A qualidade de ser simples ou não-simples
- A qualidade de ser vazio ou não-vazio
- Um retângulo de limite mínimo ou envelope da geometria
- Dimensão
- O identificador do sistema de referência espacial ao qual uma geometria está associada

Tipo

Cada geometria pertence a um tipo na hierarquia de geometrias suportadas pelo DB2 Spatial Extender. Para obter uma descrição da hierarquia de geometrias, consulte "Geometrias" na página 11. Sete tipos na hierarquia—pontos, cadeias de linhas, polígonos, coleções de geometrias, multipontos, cadeias multilinha e multipolígonos—podem ser definidos com valores de coordenadas específicos.

Coordenadas Geométricas

Todas as geometrias incluem no mínimo uma coordenada X e uma Y, a não ser que sejam geometrias vazias. Nesse caso, não conterão coordenadas. Além disso, uma geometria pode incluir uma ou mais coordenadas Z e coordenadas M. As coordenadas X, Y, Z e M são representadas como números de precisão dupla. Isto é explicado nas seguintes subseções:

- Coordenadas X e Y
- Coordenadas Z
- Coordenadas M

Coordenadas X e Y

Um *valor da coordenada X* indica um local que é relativo a um ponto de referência a leste ou oeste. Um *valor da coordenada Y* indica um local que é relativo a um ponto de referência ao norte ou sul.

Coordenadas Z

Algumas figuras geométricas apresentam uma altitude ou profundidade associada. Cada um dos pontos que formam a figura geométrica de um recurso podem incluir uma coordenada Z opcional que representa uma altitude ou profundidade normal à superfície da terra.

Coordenadas M

Uma coordenada M (medida) é um valor que transmite informações sobre um recurso geográfico e que é armazenada junto com as coordenadas que definem a localização do recurso. Por exemplo, suponha que você esteja representando estradas em seu aplicativo. Se desejar que seu aplicativo processe valores que indicam distâncias lineares ou marcos divisórios, você pode armazenar estes valores junto com as coordenadas que definem localizações na rodovia. As coordenadas M são representadas como números de precisão dupla.

Interior, Limite e Exterior

Todas as geometrias ocupam uma posição no espaço, definida por seus interiores, limites e exteriores. O exterior de uma figura geométrica é todo o espaço não ocupado pela figura geométrica. O limite de uma figura geométrica serve como a interface entre seu interior e exterior. O interior é o espaço ocupado pela figura geométrica.

Simples ou Não-simples

Os valores de alguns subtipos de geometrias (cadeias de linhas, multipontos e cadeias multilinha) são simples ou não simples. Uma geometria é simples se ela estiver de acordo com todas as regras de topologia impostas ao seu subtipo e não simples se não estiver. Uma cadeia de linhas é simples se não fizer interseção com seu interior. Um multiponto é simples se nenhum de seus elementos ocupar o mesmo espaço da coordenada. Pontos, superfícies, multisuperfícies e geometrias vazias são sempre simples.

Fechado

Uma curve será fechada se seus pontos inicial e final forem iguais. Uma multicurva será fechada se todos os seus elementos forem fechados. Um anel é uma curve simples fechada.

Vazia ou Não-vazia

Uma figura geométrica é vazia se não possuir pontos. O envelope, o limite, o interior e o exterior de uma geometria vazia não estão definidos e serão representados como nulos. Uma geometria vazia é sempre simples. Os polígonos e multipolígonos vazios têm uma área de valor 0.

MBR (Minimum Bounding Rectangle)

O MBR de uma geometria é a geometria de limite formada pelas coordenadas mínima e máxima (X,Y). Exceto para os seguintes casos especiais, os MBRs de geometrias formam um retângulo de limite:

- O MBR de qualquer ponto é o próprio ponto, porque suas coordenadas X mínima e máxima são iguais e suas coordenadas Y mínima e máxima são iguais.
- O MBR de uma cadeia de linhas horizontal ou vertical é uma cadeia de linhas representada pelo limite (os pontos extremos) da cadeia de linhas de origem.

Dimensão

Uma geometria pode ter uma dimensão de –1, 0, 1 ou 2. As dimensões são listadas conforme a seguir:

–1 É vazia

0 Não tem comprimento e uma área de valor 0 (zero)

1 Tem um comprimento maior do que 0 (zero) e uma área de valor 0 (zero)

2 Tem uma área maior do que 0 (zero)

Os subtipos de ponto e multiponto têm uma dimensão zero. Os pontos representam recursos dimensionais que podem ser modelados com uma única tupla de coordenadas, enquanto os subtipos de multiponto representam dados que devem ser modelados com um conjunto de pontos.

Os subtipos cadeia de linhas e multicadeias de linhas têm dimensão um. Elas armazenam segmentos de rodovia, extensões de rios e quaisquer outros recursos que sejam lineares na natureza.

Os subtipos polígono e multipolígono possuem uma dimensão de dois. Os recursos cujo perímetro abrange uma área definível, como florestas, terrenos e lagos, podem ser representados pelo tipo de dados polígono ou multipolígonos.

Identificador do Sistema de Referência Espacial

O identificador numérico para um sistema de referência espacial determina qual sistema de referência espacial será utilizado para representar a geometria.

Todos os sistemas de referência espacial conhecidos no banco de dados podem ser acessados por meio da exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS.

Capítulo 3. Como Utilizar o DB2 Spatial Extender

Como Utilizar o DB2 Spatial Extender

O suporte e a utilização do DB2® Spatial Extender envolvem duas atividades principais: configuração do DB2 Spatial Extender e trabalho em projetos que utilizam dados espaciais. Este tópico apresenta as interfaces que podem ser utilizadas para executar tarefas espaciais.

Interfaces para o DB2 Spatial Extender e Funcionalidade Associada

Várias interfaces permitem instalar o DB2 Spatial Extender e criar projetos que utilizam dados espaciais. Estas interfaces são:

- O Centro de Controle do DB2, uma interface gráfica do usuário que inclui janelas, blocos de notas e opções de menu com suporte ao DB2 Spatial Extender.
- Um CLP (Processador da Linha de Comandos) fornecido pelo DB2 Spatial Extender. Ele é chamado de *CLP do db2se.*
- Programas aplicativos que chamam procedimentos armazenados do DB2 Spatial Extender.

Outras interfaces permitem gerar informações espaciais. Elas incluem:

- Consultas SQL que você envia a partir do CLP do DB2, a partir de uma janela de consulta no Centro de Controle do DB2 ou a partir de um programa aplicativo.
- Ferramentas de visualização que processam informações espaciais em formato gráfico. Um exemplo é o ArcExplorer para DB2, criado pelo ESRI (Environmental Systems Research Institute) para a IBM®. O ArcExplorer para DB2 pode ser transferido por download a partir do Web site do DB2 Spatial Extender:

  http://www.ibm.com/software/data/spatial/

Tarefas Executadas para Instalar o DB2 Spatial Extender e Criar Projetos

Esta seção fornece uma visão geral das tarefas executadas para configurar o DB2 Spatial Extender e trabalhar com projetos que utilizam dados espaciais. Ela inclui um cenário que ilustra as tarefas. As tarefas estão em duas categorias:

- Configurando o DB2 Spatial Extender
- Criando projetos que utilizam dados espaciais

Configurando o DB2 Spatial Extender:

Esta seção lista as tarefas que são executadas para instalar o DB2 Spatial Extender e utiliza um cenário para ilustrar como uma empresa fictícia pode abordar cada tarefa.

Para configurar o DB2 Spatial Extender:

1. Faça planos e preparações (decida quais projetos criar, qual interface ou quais interfaces utilizar, selecione uma equipe para administrar o DB2 Spatial Extender e criar os projetos e outras coisas).

Cenário: O ambiente de sistemas de informações da Companhia de Seguros Imobiliários Porto Seguro inclui um sistema DB2 Universal Database™ e um sistema de arquivos separado somente para dados espaciais. Até certo ponto, os resultados das consultas podem incluir combinações de dados dos dois sistemas. Por exemplo, uma tabela do DB2 armazena informações sobre rendimentos e um arquivo no sistema de arquivos contém as localizações das filiais da empresa. Portanto, é possível saber quais escritórios apresentam rendimentos de quantias especificadas e, então, determinar onde estes escritórios estão localizados. Mas os dados dos dois sistemas não podem ser integrados (por exemplo, os usuários não podem unir colunas com registros do sistema de arquivos e os serviços do DB2, tais como otimização de consultas, não ficam disponíveis para o sistema de arquivos.) Para superar estas desvantagens, a Porto Seguro adquire o DB2 Spatial Extender Versão 8 e estabelece um novo departamento de Desenvolvimento Espacial (chamado de departamento Espacial, para abreviar).

A primeira missão do departamento Espacial é incluir o DB2 Spatial Extender no ambiente DB2 da Porto Seguro:

- A equipe de gerenciamento do departamento designa uma equipe de administração espacial para instalar e implementar o DB2 Spatial Extender e uma equipe de análise espacial para gerar e analisar informações espaciais.
- Como a equipe de administração tem um profundo conhecimento do UNIX®, ela decide utilizar o CLP do db2se para administrar o DB2 Spatial Extender.
- Como as decisões de negócios da Porto Seguro são conduzidas principalmente pelas exigências dos clientes, a equipe de gerenciamento decide instalar o DB2 Spatial Extender no banco de dados que contém as informações sobre seus clientes. Grande parte das informações é armazenada em uma tabela chamada CLIENTES.

2. Instale o DB2 Spatial Extender.

   Cenário: A equipe de administração espacial instala o DB2 Spatial Extender em uma máquina UNIX em um ambiente do DB2.

3. Se você tiver o DB2 Spatial Extender Versão 7, migre seus dados espaciais para o DB2 Versão 8.

   Cenário: O release da Versão 8 é o primeiro adquirido pela Porto Seguro. Nenhuma migração é necessária.

4. Configure seu banco de dados para acomodar dados espaciais. Ajuste os parâmetros de configuração para certificar que seu banco de dados tem memória e espaço suficientes para funções espaciais, arquivos de log e aplicativos do DB2 Spatial Extender.

   Cenário: Um membro da equipe de administração espacial ajusta as as características do log de transação, o tamanho de heap do aplicativo e o tamanho de heap de controle do aplicativo aos valores adequados para os requisitos do DB2 Spatial Extender.

5. Configure os recursos espaciais para seu banco de dados. Estes recursos incluem um catálogo do sistema, tipos de dados espaciais, funções espaciais, um geocoder e outros objetos. A tarefa de configuração destes recursos é referida como *ativando do banco de dados para operações espaciais.*

   O geocoder fornecido pelo DB2 Spatial Extender converte os endereços dos Estados Unidos em dados espaciais. Ele é chamado de DB2SE_USA_GEOCODER. Sua organização e outras podem fornecer geocoders que convertem endereços de fora dos Estados Unidos e outros tipos de dados em dados espaciais.

   Cenário: A equipe de administração espacial configura os recursos que serão requeridos pelos projetos que ela está planejando.

- Um membro da equipe emite um comando para obter os recursos que ativam o banco de dados para operações espaciais. Estes recursos incluem o catálogo do DB2 Spatial Extender, tipos de dados espaciais, funções espaciais e outros.
- Como a Porto Seguro está começando a expandir seus negócios para o Canadá, a equipe de administração espacial começa a solicitar aos fornecedores canadenses geocoders que convertem os endereços canadenses em dados espaciais. A Porto Seguro ainda não espera adquirir estes geocoders por alguns meses. Portanto, as primeiras localizações nas quais os dados serão coletados estarão nos Estados Unidos.

Criando Projetos que Utilizam Dados Espaciais:

Depois de instalar o DB2 Spatial Extender, você está pronto para realizar projetos que utilizem dados espaciais. Esta seção lista as tarefas envolvidas na criação de um projeto e continua o cenário no qual a Companhia de Seguros Imobiliários Porto Seguro procura integrar dados de negócios e espaciais.

Para criar um projeto que utiliza dados espaciais:

1. Faça planos e preparações (defina metas para o projeto, decida as tabelas e dados necessários, determine o sistema ou sistemas de coordenadas a serem utilizados e outras coisas).

   Cenário: O departamento Espacial se prepara para desenvolver um projeto; por exemplo:

   - A equipe de gerenciamento define essas metas para o projeto:
     - Determinar onde estabelecer novas filiais
     - Ajustar prêmios com base na proximidade dos clientes às áreas de risco (áreas com altas taxas de acidentes de tráfego, áreas com altas taxas de criminalidade e zonas de enchentes, terremotos e outros)
   - Este projeto em particular estará relacionado aos clientes e escritórios nos Estados Unidos. Assim, a equipe de administração espacial decide:
     - Utilizar um sistema de coordenadas para os Estados Unidos fornecido pelo DB2 Spatial Extender. Ele é chamado de GCS_NORTH_AMERICAN_1983.
     - Utilizar DB2SE_USA_GEOCODER, porque ele foi projetado para efetuar o geocode de endereços dos Estados Unidos.
   - A equipe de administração espacial decide quais os dados necessários para satisfazer as metas do projeto e em que tabelas estes dados ficarão contidos.
2. Se necessário, crie um sistema de coordenadas.

   Cenário: Como a Safe Harbor optou por utilizar GCS_NORTH_AMERICAN_1983, a empresa pode ignorar esta etapa.
3. Decida se um sistema de referência espacial existente atende às suas necessidades. Se nenhum atender, crie um.

   Um *sistema de referência espacial* é um conjunto de valores de parâmetros que inclui:

   - Coordenadas que definem a máxima extensão possível de espaço referido por um determinado intervalo de coordenadas. É necessário determinar o intervalo máximo possível de coordenadas que podem ser determinadas a partir do sistema de coordenadas que está sendo utilizado e selecionar ou criar um sistema de referência espacial que reflita este intervalo.
   - O nome do sistema de coordenadas a partir do qual as coordenadas são derivadas.

- Números utilizados em operações matemáticas para converter coordenadas recebidas como entrada em valores que podem ser processados com eficiência máxima. As coordenadas são armazenadas em seu formato convertido e retornadas ao usuário em seu formato original.

Cenário: O DB2 Spatial Extender fornece um sistema de referência espacial, NAD83_SRS_1, desenvolvido para ser utilizado com o GCS_NORTH_AMERICAN_1983. A equipe de administração espacial decide utilizar o NAD83_SRS_1.

4. Crie colunas espaciais conforme necessário. Observe que, em muitos casos, se os dados em uma coluna espacial tiverem que ser lidos por uma ferramenta de visualização, a coluna deverá ser a única coluna espacial na tabela ou exibição à qual ela pertence. Como alternativa, se a coluna for uma de várias colunas espaciais em uma tabela, ela poderá ser incluída em uma exibição que não tenha outras colunas espaciais e as ferramentas de visualização poderão ler os dados a partir desta exibição.

   Cenário: A equipe de administração espacial define colunas para conter dados espaciais.

   - A equipe acrescenta uma coluna LOCALIZAÇÃO na tabela CLIENTES. A tabela já contém endereços dos clientes. O DB2SE_USA_GEOCODER irá convertê-los em dados espaciais. Em seguida, o DB2 armazenará esses dados na coluna LOCALIZAÇÃO.
   - A equipe cria uma tabela LOCALIZAÇÕES_DOS_ESCRITÓRIOS e uma tabela VENDAS_DO_ESCRITÓRIO para conter dados que agora estão armazenados no sistema de arquivos separado. Estes dados incluem os endereços das filiais da Porto Seguro, os dados espaciais originados destes endereços através de um geocoder, e dados espaciais que definem uma zona dentro de um raio de cinco milhas ao redor de cada escritório. Os dados derivados do geocoder serão colocados em uma coluna LOCALIZAÇÃO na tabela LOCALIZAÇÕES_DOS_ESCRITÓRIOS e os dados que definem as regiões serão colocados em uma coluna ÁREA_DE_VENDAS na tabela VENDAS_DO_ESCRITÓRIO.

5. Configure colunas espaciais para serem acessadas por ferramentas de visualização, conforme necessário. Faça isto registrando as colunas no catálogo do DB2 Spatial Extender. Quando registrar uma coluna espacial, o DB2 Spatial Extender impõe uma limitação de que todos os dados na coluna devem pertencer ao mesmo sistema de referência espacial. Esta limitação garante a integridade dos dados — uma exigência da maioria das ferramentas de visualização.

   Cenário: A equipe de administração espacial espera utilizar ferramentas de visualização para processar o conteúdo das colunas LOCALIZAÇÃO e da coluna ÁREA_DE_VENDAS graficamente em um mapa. Portanto, a equipe registra todas as três colunas.

6. Preencha as colunas espaciais:

   Para um projeto que requer que os dados espaciais sejam importados, importe-os.

   Para um projeto que requer um geocoder:

   - Defina antecipadamente as informações de controle necessárias quando um geocoder é chamado.
   - Como opção, configure o geocoder para ser executado automaticamente sempre que um novo endereço for incluído no banco de dados, ou sempre que um endereço existente for atualizado.

   Execute o geocoder no modo batch, conforme necessário.

Para um projeto que requer que os dados espaciais sejam criados por uma função espacial, execute esta função.

Cenário: A equipe de administração espacial preenche a coluna LOCALIZAÇÃO da tabela CLIENTE, a tabela LOCALIZAÇÕES_DOS_ESCRITÓRIOS, a tabela VENDAS_DO_ESCRITÓRIO e uma nova tabela ZONAS_DE_PERIGO:

- A equipe utiliza o DB2SE_USA_GEOCODER para efetuar o geocode dos endereços na tabela CLIENTE. As coordenadas geradas pelo geocoding são inseridas na coluna LOCALIZAÇÃO da tabela.
- A equipe usa um utilitário para carregar os dados do escritório do sistema de arquivos para um arquivo. Em seguida, a equipe importa esses dados para a nova tabela LOCALIZAÇÕES_DOS_ESCRITÓRIOS.
- A equipe cria uma tabela ZONAS_DE_PERIGO, registra suas colunas espaciais e importa dados para ela. Os dados se originam de um arquivo adquirido de um fornecedor de mapas.

7. Se necessário, facilite o acesso às colunas espaciais. Isto envolve a definição de índices que permitem que o DB2 acesse dados espaciais rapidamente e a definição de exibições que permitem que os usuários recuperem dados inter-relacionados de forma eficiente. Se deseja que as ferramentas de visualização acessem as colunas espaciais das exibições, será necessário registrar essas colunas no DB2 Spatial Extender também.

   Cenário: A equipe de administração espacial cria índices para as colunas registradas. Ela cria, então, uma exibição que junta as colunas das tabelas CLIENTES e ZONAS_DE_PERIGO. Em seguida, registra as colunas espaciais nessa exibição.

8. Gere informações espaciais e informações de negócios relacionadas. Analise as informações. Esta tarefa envolve a consulta a colunas espaciais e colunas não espaciais relacionadas. Em tais consultas, você pode incluir funções do DB2 Spatial Extender que retornam uma grande variedade de informações; por exemplo, coordenadas que definem uma zona de segurança proposta em torno de um depósito de lixo tóxico ou a distância mínima entre esse local e a área de casas mais próxima.

   Cenário: A equipe de análise espacial executa consultas para obter informações que irão ajudar a alcançar suas metas originais: determinar onde estabelecer novas filiais e ajustar os prêmios com base na proximidade dos clientes às áreas de risco.

Tarefas Relacionadas:

- "Configurando e Instalando o Spatial Extender" na página 25
- "Ativando um Banco de Dados para Operações Espaciais" na página 54
- "Registrando um Geocoder" na página 56
- "Configurando um Banco de Dados para Acomodar Dados Espaciais" na página 49
- "Importando Dados de Formas para uma Tabela Nova ou Existente" na página 88
- "Importando Dados de Transferência SDE para uma Tabela Nova ou Existente" na página 90
- "Configurando Operações de Geocoding" na página 95
- "Configurando um Geocoder para Execução Automática" na página 97
- "Executando um Geocoder no Modo Batch" na página 98
- "Exportando Dados para um Arquivo de Transferência SDE" na página 92

- "Selecionando ou Criando Sistemas de Coordenadas" na página 65
- "Registrando Colunas Espaciais" na página 85
- "Criando Colunas Espaciais" na página 83
- "Criando Índices de Grades Espaciais" na página 108
- "Chamando Procedimentos Armazenados do DB2 Spatial Extender a Partir de um Aplicativo" na página 136
- "Incluindo o arquivo de cabeçalho do DB2 Spatial Extender em aplicativos espaciais" na página 135

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127

Parte 2. Configurando o DB2 Spatial Extender

Capítulo 4. Informações Iniciais do DB2 Spatial Extender

Este capítulo fornece instruções para instalar e configurar o Spatial Extender para AIX, HP-UX, Windows NT, Windows 2000, Linux, Linux para z/OS e Solaris Operating Environments. Este capítulo também explica como resolver alguns problemas de instalação e configuração que possam surgir quando você chamar o Spatial Extender.

Configurando e Instalando o Spatial Extender—Etapas

Esta seção contém detalhes para os seguintes tópicos:

- Requisitos do sistema para instalar o Spatial Extender
- Instruções de instalação para as plataformas UNIX e Windows
- Explicação sobre como criar um ambiente de instância do DB2 Spatial Extender
- Verificando a instalação do Spatial Extender
- Dicas de resolução de problemas para o programa de amostra de instalação

Configurando e Instalando o Spatial Extender

Um sistema DB2 Spatial Extender consiste no DB2 Universal Database, DB2 Spatial Extender e, para a maioria dos aplicativos, um geonavegador. Um geonavegador não é requerido, mas é útil para processar visualmente os resultados de consultas espaciais, geralmente em forma de mapas. Os bancos de dados ativados para as operações espaciais estão localizados no servidor. Você pode utilizar os aplicativos do cliente para acessar os dados espaciais por meio dos procedimentos armazenados e das consultas espaciais do DB2 Spatial Extender. Você também pode configurar o DB2 Spatial Extender em um ambiente independente, que é uma configuração na qual o cliente e o servidor residem na mesma máquina. Nas configurações cliente–servidor e independente, é possível exibir dados espaciais com um navegador geográfico, como o ArcExplorer para DB2 ou os conjuntos de ferramentas ArcGIS do ESRI em execução com o ArcSDE.

Você pode fazer download de uma cópia gratuita do ArcExplorer para DB2 do Web site do DB2 Spatial Extender da IBM:

http://www.ibm.com/software/data/spatial/

Pré-requisitos:

Antes de instalar o DB2 Spatial Extender, é necessário ter o software DB2 instalado e configurado no cliente e no servidor.

Procedimento:

1. Certifique-se de que seu sistema atenda todos os requisitos de software.
2. Instale o Spatial Extender. As etapas variam, dependendo de seu sistema operacional:
   - Windows
   - AIX
   - HP-UX
   - Solaris Operating Environment

- Linux

3. Para plataformas UNIX: Crie um ambiente de instância do Spatial Extender.
4. Verifique a instalação
5. Se necessário, consulte as dicas para resolução de problemas e execute as ações apropriadas para corrigir quaisquer problemas.
6. Se desejar acessar a documentação do DB2 em seu computador e ainda não tiver instalado o Centro de Informações do DB2, consulte Instalando o Centro de Informações do DB2 (UNIX)″ ou Instalando o Centro de Informações do DB2 (Windows). O Centro de Informações do DB2 contém a documentação para o DB2 Universal Database e produtos relacionados do DB2.

Conceitos Relacionados:

- "Requisitos do Sistema para Instalar o Spatial Extender" na página 26

Tarefas Relacionadas:

- "Instalando o DB2 Spatial Extender para Windows" na página 27
- "Instalando o DB2 Spatial Extender para AIX" na página 29
- "Instalando o DB2 Spatial Extender para HP-UX" na página 31
- "Instalando o DB2 Spatial Extender para Solaris Operating Environment" na página 33
- "Instalando o DB2 Spatial Extender para Linux" na página 35
- "Criando o Ambiente da Instância do DB2 Spatial Extender" na página 37
- "Verificando a instalação do Spatial Extender" na página 39
- "Resolução de Problemas de Instalação" na página 40
- "Installing the DB2 Information Center (UNIX)" no *Infrastructure Topics (DB2 Common Files)*
- "Installing the DB2 Information Center using the DB2 Setup wizard (Windows)" no *Infrastructure Topics (DB2 Common Files)*
- "Fazendo Download do ArcExplorer para DB2" na página 41

Referência Relacionada:

- "CDs para Dados e Mapas do DB2 Spatial Extender" na página 42

Requisitos do Sistema para Instalar o Spatial Extender

Antes de instalar o DB2® Spatial Extender, assegure que o sistema atenda a todos os requisitos de software e de espaço em disco descritos a seguir.

Sistemas Operacionais:

Você pode instalar o DB2 Spatial Extender em versões de 32 bits e 64 bits do Windows®, AIX®, HP-UX, Solaris Operating Environment e Linux para Intel. O Spatial Extender suporta o Linux para S/390® (32 bits), mas não suporta o Linux para zSeries® (64 bits).

Requisitos de Software:

Para instalar o Spatial Extender, você deve ter o seguinte software DB2 instalado e configurado no servidor:

Software Servidor
DB2 Universal Database™ Enterprise Server Edition Versão 8.2deve estar

instalado no sistema *antes* da instalação do DB2 Spatial Extender. Se você pretende utilizar o Centro de Controle do DB2, crie e configure o DAS (DB2 Administration Server). Para obter mais informações sobre como criar e configurar o DAS, consulte o *IBM® DB2 Universal Database Administration Guide: Implementation*.

Software do Cliente Espacial
Se você instalar o DB2 Spatial Extender no Windows, a instalação padrão do Spatial Extender incluirá o cliente espacial. Solaris Operating EnvironmentPara o DB2 Spatial Extender no AIX, HP-UX, Solaris Operating Environment, Linux para Intel ou Linux para S/390, o cliente espacial pode ser instalado quando o servidor DB2 for instalado com o cliente de desenvolvimento de administração ou do aplicativo. Se esses clientes não forem instalados, você deve instalar o cliente espacial manualmente, escolhendo a opção de instalação Personalizada.

Requisitos de Espaço em Disco:

Para instalar o Spatial Extender, o sistema deve atender aos seguintes requisitos de espaço em disco listados na tabela a seguir. O código da biblioteca do DB2 Spatial Extender integra o código para o DB2 Geodetic Extender mas não a chave de licença do Geodetic.

Tabela 1. Requisitos de Espaço em Disco para o DB2 Spatial Extender

Software DB2 Spatial Extender	Espaço em disco
Software Servidor para o DB2 Spatial Extender:	Espaço Total em Disco de 596 MB:
• Código da biblioteca do servidor, dados de referência do geocoder de amostra e documentação do Spatial Extender e do Geodetic Extender	• 33 MB
• Opcional e disponível em um CD separado: dados de referência de geocoder (Estados Unidos)	• 563 MB

Tabela 1 especifica o espaço em disco necessário para a instalação do DB2 Universal Database e do DB2 Spatial Extender em uma instalação típica para Windows ou com componentes pré-selecionados no AIX, HP-UX, Solaris Operating Environment, Linux para Intel e Linux para S/390. Se você estiver instalando o DB2 Spatial Extender ou tiver instalado o DB2 Universal Database com um tipo de instalação diferente, os cálculos de espaço em disco serão diferentes.

Quando o sistema atende a todos os requisitos de software e de espaço em disco, você pode instalar o Spatial Extender.

Instalando o DB2 Spatial Extender para Windows

Esta tarefa faz parte da tarefa principal de Configurando o DB2 Spatial Extender.

Você pode instalar o DB2 Spatial Extender em sistemas operacionais Windows utilizando o assistente do DB2 Setup ou um arquivo de resposta.

Recomendação: utilize o assistente do DB2 Setup para instalar o Spatial Extender. O assistente de instalação oferece uma interface gráfica fácil de utilizar com ajuda de instalação, criação automatizada de usuários e grupos, configuração de protocolos e criação de instâncias.

Se estiver utilizando o assistente do DB2 Setup para instalar o Spatial Extender, você poderá clicar em **Cancelar** a qualquer momento durante a instalação para sair do processo.

Pré-requisitos:

Antes de instalar o produto DB2 Spatial Extender, você deve ter um produto do servidor DB2 Versão 8 instalado.

Procedimento:

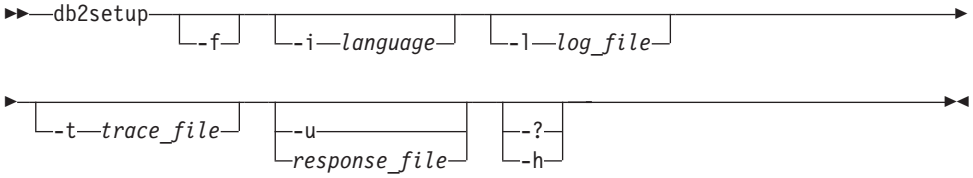
Para instalar o Spatial Extender para Windows utilizando o assistente do DB2 Setup:

1. Insira o CD do Spatial Extender na unidade de CD. O DB2 Setup Launchpad, é uma interface a partir da qual você pode instalar o DB2 Spatial Extender.
2. Clique em **Instalar produtos**.
3. Selecione o DB2 Spatial Extender como o produto que deseja instalar e clique em **AVANÇAR**. É lançado o assistente do DB2 Setup. Clique em **AVANÇAR**. Utilize o assistente do DB2 Setup para orientá-lo na instalação e nas etapas de instalação restantes. A qualquer momento, durante a instalação, você pode clicar em **Ajuda** para lançar a ajuda on-line da instalação.

Para instalar o Spatial Extender para Windows utilizando o arquivo de resposta:

1. Efetue o logon no sistema com a conta de usuário a ser usada para fazer a instalação.
2. Insira o CD do Spatial Extender. Consulte o *DB2 Installation and Configuration Supplement* para informações adicionais.
3. Execute o programa de instalação emitindo o seguinte a partir de um prompt de comandos:

Comando db2setup



Em que:

-f Força a parada dos processos do DB2 antes da instalação.

-i (*language***)**
É o código de idioma de duas letras do idioma no qual será executada a instalação.

-l (*log_file* **)**
É o caminho completo e o nome do arquivo do arquivo de log a ser utilizado.

-t (*trace_file***)**
Gera um arquivo completo com informações de rastreio da instalação.

-u (*response_file***)**
Especifica o nome completo do arquivo de resposta. Se você alterou e renomeou a amostra do arquivo de resposta que foi fornecido, certifique-se de que este parâmetro corresponda ao novo nome. Este parâmetro é necessário. O arquivo de resposta está localizado em db2\Windows\samples\db2gse.rsp no CD de instalação do DB2 Spatial Extender.

-?, -h Gera informações de uso.

4. Após a instalação, verifique as mensagens no arquivo de log.

Conceitos Relacionados:

- "Requisitos do Sistema para Instalar o Spatial Extender" na página 26

Tarefas Relacionadas:

- "Creating and editing a response file (Windows)" na publicação *Suplemento de Instalação e Configuração*
- "Verificando a instalação do Spatial Extender" na página 39
- "Resolução de Problemas de Instalação" na página 40

Instalando o DB2 Spatial Extender para AIX

Você pode instalar o DB2 Spatial Extender para AIX utilizando o assistente do DB2 Setup, utilizando o script db2_install ou utilizando a SMIT (System Management Interface Tool).

Recomendação: Utilize o assistente do DB2 Setup para instalar o Spatial Extender. O assistente para Instalação oferece uma interface gráfica fácil de utilizar com ajuda de instalação, criação automatizada de usuários e grupos, configuração de protocolos e criação de instâncias. Se você optar por não utilizar o assistente, poderá instalar o Spatial Extender utilizando o script db2_install ou utilizando a SMIT (System Management Interface Tool) do AIX. O uso da ferramenta SMIT para instalar o Spatial Extender é recomendado apenas para usuários avançados em situações em que há necessidade de maior controle manual sobre o processo de configuração.

Pré-requisitos:

Antes de instalar o Spatial Extender no AIX:

- Certifique-se de que seu sistema atenda a todos os requisitos de software, memória e espaço em disco.
- Atualize os parâmetros de configuração e reinicie o sistema para todos os clientes e servidores DB2 no AIX.
- Você deve ter um produto do servidor DB2 Versão 8 instalado se estiver instalando em um servidor ou em um ambiente independente.

Nota: Verifique se o Cliente DB2 Spatial já está instalado.O cliente Spatial Extender e os componentes de amostra estão disponíveis com o cliente e o servidor DB2. Você pode instalar estes componentes espaciais quando utilizar o tipo de instalação personalizada do DB2 e selecionar o recurso **Cliente Spatial Extender** em **Suporte ao Cliente** e selecionar o recurso **Amostras do Spatial Extender** em **Ferramentas de Desenvolvimento de**

Aplicativos. Se precisar apenas da funcionalidade do cliente espacial e já tiver instalado esses componentes espaciais com o DB2, não precisará executar o procedimento de instalação do DB2 Spatial Extender indicado a seguir.

- Você deve ter autoridade root.

Procedimento:

Para instalar o Spatial Extender utilizando o assistente do DB2 Setup:

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD do Spatial Extender. O DB2 Setup Launchpad, é uma interface a partir da qual você pode instalar o DB2 Spatial Extender. Para obter informações sobre como montar um CD, consulte o *DB2 Installation and Configuration Supplement*.
3. Selecione o DB2 Spatial Extender como o produto que deseja instalar e clique em **AVANÇAR**.
4. É aberta a janela do assistente do DB2 Setup. Utilize o assistente do DB2 Setup para orientá-lo na configuraçao e nas etapas de instalação restantes. A qualquer momento, durante a instalação, você pode clicar em **Ajuda** para lançar a ajuda on-line da instalação.

Para instalar o DB2 Spatial Extender utilizando o script db2_install:

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD apropriado.
3. Digite o comando **./db2_install** para iniciar o script db2_install. O script db2_install pode ser localizado no diretório raiz no CD do produto DB2 Versão 8. O script db2_install solicita a palavra-chave do produto.
4. Digite **DB2.GSE** para instalar o DB2 Spatial Extender.

Para instalar o DB2 Spatial Extender utilizando a SMIT (System Management Interface Tool):

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD do Spatial Extender.
3. Digite o comando **smit install_latest**.
4. Digite `/cdrom/db2` no **dispositivo/diretório de ENTRADA** para o campo software.
5. Clique em **DO** ou pressione Enter para verificar se o diretório de instalação existe.
6. No campo **Software a ser instalado**, identifique se devem ser instalados os componentes cliente ou servidor.

   Nota: Consulte o arquivo ComponentList.htm no CD do DB2 Spatial Extender para obter uma lista completa dos componentes que devem ser instalados no DB2 Spatial Extender.
7. Clique em **DO** ou pressione Enter. Será solicitado que confirme os parâmetros de instalação.
8. Pressione Enter para confirmar.
9. Efetue logout.

Quando a instalação estiver concluída, o Spatial Extender será instalado no diretório /usr/opt/db2_08_01 juntamente com outros produtos do DB2.

Depois de instalar o Spatial Extender, crie seu ambiente de instância do DB2 se ainda não o criou e, em seguida, verifique a instalação.

Conceitos Relacionados:

- "Requisitos do Sistema para Instalar o Spatial Extender" na página 26

Tarefas Relacionadas:

- "Installing a DB2 product using SMIT (AIX)" na publicação *Suplemento de Instalação e Configuração*
- "Mounting the CD-ROM (AIX)" na publicação *Quick Beginnings for DB2 Servers*
- "Installing a DB2 product using the db2_install script (UNIX)" na publicação *Suplemento de Instalação e Configuração*
- "Verificando a instalação do Spatial Extender" na página 39
- "Resolução de Problemas de Instalação" na página 40

Instalando o DB2 Spatial Extender para HP-UX

Você pode instalar o Spatial Extender utilizando o assistente do DB2® Setup, utilizando o script db2_install ou utilizando o comando **swinstall**.

Recomendação: utilize o assistente do DB2 Setup para instalar o Spatial Extender. O assistente de instalação oferece uma interface gráfica fácil de utilizar com ajuda de instalação, criação automatizada de usuários e grupos, configuração de protocolos e criação de instâncias. Se você optar por não utilizar o assistente, poderá instalar o Spatial Extender para HP-UX utilizando o script db2_install ou utilizando o comando **swinstall**. O uso do comando **swinstall** do HP-UX para instalar o Spatial Extender é recomendado apenas para usuários avançados em situações em que há necessidade de maior controle manual sobre o processo de configuração.

Pré-requisitos:

Antes de instalar o produto DB2 Spatial Extender para HP-UX:

- Certifique-se de que seu sistema atenda a todos os requisitos de hardware, software e memória.
- Você deve ter um produto do servidor DB2 Versão 8 instalado.

  Nota: Verifique se o Cliente DB2 Spatial já está instalado.O cliente Spatial Extender e os componentes de amostra estão disponíveis com o cliente e o servidor DB2. Você pode instalar estes componentes espaciais quando utilizar o tipo de instalação personalizada do DB2 e selecionar o recurso **Cliente Spatial Extender** em **Suporte ao Cliente** e selecionar o recurso **Amostras do Spatial Extender** em **Ferramentas de Desenvolvimento de Aplicativos**. Se precisar apenas da funcionalidade do cliente espacial e já tiver instalado esses componentes espaciais com o DB2, não precisará executar o procedimento de instalação do DB2 Spatial Extender indicado a seguir.
- Atualize os parâmetros de configuração e reinicie o sistema para todos os clientes e servidores DB2 no HP-UX.
- Você deve ter autoridade root.

Procedimento:

Para instalar o Spatial Extender para HP-UX utilizando o assistente do DB2 Setup:

1. Insira e monte o CD do DB2 Spatial Extender. O DB2 Setup Launchpad, é uma interface a partir da qual você pode instalar o DB2 Spatial Extender.
2. Selecione o DB2 Spatial Extender como o produto que deseja instalar e clique em **AVANÇAR**. É lançado o assistente do DB2 Setup. Clique em **AVANÇAR**. Utilize o assistente do DB2 Setup para orientá-lo na instalação e nas etapas de instalação restantes. A qualquer momento, durante a instalação, você pode clicar em **Ajuda** para lançar a ajuda on-line da instalação.

Para instalar o Spatial Extender para HP-UX utilizando o script db2_install:

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD apropriado.
3. Digite o comando **./db2_install** para iniciar o script db2_install. O script db2_install pode ser localizado no diretório raiz no CD do produto DB2 Versão 8. O script db2_install solicita a palavra-chave do produto.
4. Digite **DB2.GSE** para instalar o DB2 Spatial Extender.

Para instalar o Spatial Extender para HP-UX utilizando o comando **swinstall**:

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD do Spatial Extender.
3. Execute o programa swinstall utilizando o seguinte comando:

   ```
   swinstall -x autoselect_dependencies=true
   ```

   Este comando abre a janela Seleção de Software e a janela Especificar Origem. Se necessário, altere o valor no campo **Nome do host de origem** na janela Especificar Origem.
4. No campo **Caminho do Depósito de Origem**, insira */cdrom*/db2/hpux, em que */cdrom* representa o diretório de montagem do CD.
5. Clique em **OK** para retornar à janela Seleção de Software. A janela Seleção de Software contém uma lista de softwares disponíveis para instalação.
6. Selecione os produtos dos quais você tem licença para instalar.
7. Selecione **Marcar para Instalação** a partir do menu **Ações** para escolher o produto a ser instalado. Aparece uma mensagem:

   ```
   Além do software que você acabou de marcar, outro software estava
   automaticamente marcado para resolver dependências. Esta mensagem
   não aparecerá novamente.
   ```
8. Selecione **OK**.
9. Selecione **Instalar (análise)** a partir do menu **Ações** para iniciar a instalação do produto e abrir a janela Análise de Instalação.
10. Selecione **OK** na janela Análise de Instalação quando o campo **Status** exibir uma mensagem de Pronto.
11. Selecione **Sim** nas janelas de Confirmação para confirmar se deseja instalar o software.

    Exiba a janela Instalar para ler os dados de processamento enquanto o software está sendo instalado, até que o campo **Status** indique Pronto e a janela Nota seja aberta. O programa swinstall carrega o conjunto de arquivos e executa os scripts de controle para o conjunto de arquivos.
12. Selecione **Sair** a partir do menu **Arquivo** para sair do swinstall.

Quando a instalação estiver concluída, o Spatial Extender será instalado no diretório /opt/IBM/db2/V8.1 juntamente com outros produtos do DB2.

Depois de instalar o Spatial Extender, crie seu ambiente de instância do DB2 se ainda não o criou e, em seguida, verifique a instalação.

Conceitos Relacionados:

- "Requisitos do Sistema para Instalar o Spatial Extender" na página 26

Tarefas Relacionadas:

- "Installing a DB2 product using swinstall (HP-UX)" na publicação *Suplemento de Instalação e Configuração*
- "Mounting the CD-ROM (HP-UX)" na publicação *Quick Beginnings for DB2 Servers*
- "Installing a DB2 product using the db2_install script (UNIX)" na publicação *Suplemento de Instalação e Configuração*
- "Verificando a instalação do Spatial Extender" na página 39
- "Resolução de Problemas de Instalação" na página 40

Instalando o DB2 Spatial Extender para Solaris Operating Environment

Você pode instalar o Spatial Extender utilizando o assistente do DB2® Setup, utilizando o script db2_install ou utilizando o comando **pkgadd**.

Recomendação: Utilize o assistente do DB2 Setup para instalar o DB2 Spatial Extender. O assistente de instalação oferece uma interface gráfica fácil de utilizar com ajuda de instalação, criação automatizada de usuários e grupos, configuração de protocolos e criação de instâncias. Se você optar por não utilizar o assistente, poderá instalar o Spatial Extender utilizando o script db2_install ou utilizando o comando **pkgadd** do Solaris Operating Environment. A utilização do comando **pkgadd** do Solaris Operating Environment é recomendada somente para usuários avançados em situações onde é requerido maior controle manual sobre o processo de instalação.

O DB2 Spatial Extender é composto de funções e componentes diferentes que são referidos como pacotes no Solaris Operating Environment. Ao instalar o Spatial Extender utilizando o comando **pkgadd**, você deve instalar cada pacote necessário e cada pacote associado para as funções opcionais que deseja utilizar. O arquivo ComponentList.htm no CD do DB2 Spatial Extender tem uma lista completa dos pacotes que devem ser instalados para o DB2 Spatial Extender. O arquivo ComponentList.htm está localizado em */cdrom*/db2/solaris, em que */cdrom* é o ponto de montagem para o CD do DB2 Spatial Extender.

Pré-requisitos:

Antes de instalar o produto DB2 Spatial Extender para Solaris Operating Environment:

- Certifique-se de que seu sistema atenda a todos os requisitos de hardware, software e memória.
- Você deve ter um produto do servidor DB2 Versão 8 instalado se estiver instalando em um servidor ou em um ambiente independente.

  Nota: Verifique se o Cliente DB2 Spatial já está instalado.O cliente Spatial Extender e os componentes de amostra estão disponíveis com o cliente e o servidor DB2. Você pode instalar estes componentes espaciais quando

utilizar o tipo de instalação personalizada do DB2 e selecionar o recurso **Cliente Spatial Extender** em **Suporte ao Cliente** e selecionar o recurso **Amostras do Spatial Extender** em **Ferramentas de Desenvolvimento de Aplicativos**. Se precisar apenas da funcionalidade do cliente espacial e já tiver instalado esses componentes espaciais com o DB2, não precisará executar o procedimento de instalação do DB2 Spatial Extender indicado a seguir.

- Atualize os parâmetros de configuração e reinicie o sistema para todos os clientes e servidores DB2 no Solaris Operating Environment.
- Você deve ter autoridade root.

Procedimento:

Para instalar o DB2 Spatial Extender para Solaris Operating Environments utilizando o assistente do DB2 Setup:

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD do DB2 Spatial Extender. É aberta a barra de ativação de Instalação do DB2, uma interface a partir da qual você pode instalar o DB2 Spatial Extender. Para obter informações sobre como montar um CD, consulte *DB2 for UNIX Quick Beginnings*.
3. Selecione **Spatial Extender** como o produto que deseja instalar e clique em **AVANÇAR**.
4. É lançado o assistente do DB2 Setup. Utilize o assistente do DB2 Setup para orientá-lo na instalação e nas etapas de instalação restantes. A qualquer momento, durante a instalação, você pode clicar em **AJUDA** para lançar a ajuda on-line da instalação.

Para instalar o DB2 Spatial Extender utilizando o script db2_install:

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD apropriado.
3. Digite o comando **./db2_install** para iniciar o script db2_install. O script db2_install pode ser localizado no diretório raiz no CD do produto DB2 Versão 8. O script db2_install solicita a palavra-chave do produto.
4. Digite **DB2.GSE** para instalar o DB2 Spatial Extender.

Para instalar o DB2 Spatial Extender para Solaris utilizando o comando **pkgadd** :

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD do DB2 Spatial Extender.
3. Identifique os pacotes obrigatórios e opcionais que deseja instalar. Consulte o arquivo ComponentList.htm em seu CD para obter uma lista completa dos componentes que devem ser instalados para o DB2 Spatial Extender.
4. Execute o comando **pkgadd** para cada pacote que deseja instalar, digitando:

   ```
   pkgadd package_name
   ```

   Nesse comando, *package_name* é o pacote que você deseja instalar.

   Por exemplo, se deseja instalar o Spatial Extender Base Server Support, você precisa instalar o pacote db2gssg81, digitando o seguinte comando:

   ```
   pkgadd db2gssg81
   ```

Quando a instalação estiver concluída, seu software do Spatial Extender será instalado no diretório /opt/IBM/db2/V8.1.

Crie o ambiente da instância do DB2 se ainda não criou e, em seguida, verifique a instalação.

Conceitos Relacionados:

- "Requisitos do Sistema para Instalar o Spatial Extender" na página 26

Tarefas Relacionadas:

- "Installing a DB2 product using pkgadd (Solaris Operating Environments)" na publicação *Suplemento de Instalação e Configuração*
- "Installing a DB2 product using the db2_install script (UNIX)" na publicação *Suplemento de Instalação e Configuração*
- "Mounting the CD-ROM (Solaris Operating Environment)" na publicação *Quick Beginnings for DB2 Servers*
- "Verificando a instalação do Spatial Extender" na página 39
- "Resolução de Problemas de Instalação" na página 40

Instalando o DB2 Spatial Extender para Linux

Você pode instalar o DB2 Spatial Extender para Linux utilizando o assistente do DB2 Setup, utilizando o script db2_install ou utilizando o comando **rpm**.

Recomendação: Utilize o assistente do DB2 Setup para instalar o Spatial Extender. O assistente de instalação oferece uma interface gráfica fácil de utilizar com ajuda de instalação, criação automatizada de usuários e grupos, configuração de protocolos e criação de instâncias. Se você escolher por não utilizar o assistente, poderá instalar o Spatial Extender utilizando o script db2_install ou o comando **rpm**. O uso do comando **rpm** do Linux para instalar o Spatial Extender é recomendado apenas para usuários avançados em situações em que há necessidade de maior controle manual sobre o processo de configuração.

Pré-requisitos:

Antes de instalar o produto DB2 Spatial Extender para Linux:

- Certifique-se de que seu sistema atenda a todos os requisitos de hardware, software e memória.
- Assegure-se de que possui um produto servidor DB2 Versão 8 instalado, se estiver instalando em um ambiente do servidor ou em um ambiente independente.

  Nota:

  - Verifique se o DB2 Spatial Client já está instalado. O cliente DB2 Spatial Extender e os componentes de amostra estão disponíveis com o cliente e o servidor DB2.Você pode instalar esses componentes espaciais ao utilizar o tipo de instalação personalizada do DB2 e selecionar o recurso **Cliente Spatial Extender** em **Suporte ao Cliente** e selecionar o recurso **Amostras do Spatial Extender** em **Ferramentas de Desenvolvimento de Aplicativos**. Se precisar apenas da funcionalidade do cliente espacial e já tiver instalado esses componentes espaciais com o DB2, não precisará executar o procedimento de instalação do DB2 Spatial Extender indicado a seguir.
  - Atualize os parâmetros de configuração e reinicie o sistema para todos os clientes e servidores DB2 no Linux.
  - Assegure-se de que possui autoridade root.

Procedimento:

Para instalar o DB2 Spatial Extender utilizando o assistente do DB2 Setup:

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD do DB2 Spatial Extender. É aberta a barra de ativação de Instalação do DB2, uma interface a partir da qual você pode instalar o DB2 Spatial Extender. Para obter informações sobre como montar um CD, consulte o *DB2 Installation and Configuration Supplement*.
3. Clique em **Instalar Produtos**.
4. Selecione **Spatial Extender** como o produto que deseja instalar e clique em **AVANÇAR**.
5. Selecione a opção desejada na janela do assistente do DB2 Setup. Você tem a opção de instalar o **DB2 Spatial Extender** ou o **DB2 Application Development Client**.
   - Se precisar somente da funcionalidade de cliente espacial, selecione **DB2 Application Development Client** e os seguintes recursos:
     - Recurso **Cliente Spatial Extender** em **Suporte ao Cliente**
     - Opcional: recurso **Amostras do Spatial Extender** em **Ferramentos de Desenvolvimento de Aplicativos**
   - Se precisar das funcionalidades de servidor e cliente espacial, selecione **DB2 Spatial Extender** e os seguintes recursos:
     - Recurso **Suporte ao Servidor Base do Spatial Extender** em **Suporte ao Servidor**
     - Recurso **Cliente Spatial Extender** em **Suporte ao Cliente Spatial Extender**
     - Opcional: recurso **Amostras do Spatial Extender** em **Ferramentas de Desenvolvimento de Aplicativos**

   Utilize o assistente do DB2 Setup para orientá-lo na configuraçao e nas etapas de instalação restantes. A qualquer momento, durante a instalação, você pode clicar em **Ajuda** para ativar a ajuda on-line da instalação.

Para instalar o DB2 Spatial Extender utilizando o script db2_install:

1. Efetue login como um usuário com autoridade root.
2. Insira e monte o CD apropriado.
3. Insira o comando **./db2_install** para iniciar o script db2_install. O script db2_install pode ser localizado no diretório root no produto DB2 Versão 8. O script db2_install solicitará a você a palavra-chave do produto.
4. Digite **DB2.GSE** para instalar o DB2 Spatial Extender.

Para instalar o Spatial Extender para Linux utilizando o comando **rpm**:

1. Efetue login como um usuário com autoridade root.
2. Ative seu sistema para a instalação do DB2 para Linux. Consulte o *DB2 Installation and Configuration Supplement* para informações adicionais.
3. Insira e monte o CD do DB2 Spatial Extender.
4. Identifique os pacotes obrigatórios e opcionais que deseja instalar.

   Nota: Consulte o arquivo ComponentList.htm no CD do DB2 Spatial Extender para obter uma lista completa dos componentes que devem ser instalados no DB2 Spatial Extender.
5. Execute o comando **rpm** para cada pacote que deseja instalar:

   ```
   rpm -ivh package_name
   ```

Por exemplo, para instalar o servidor, instale o pacote IBM_db2gssg81-8.1.0-0.i386.rpm, inserindo o seguinte comando:

```
rpm -IBM_db2gssg81-8.1.0-0.i386.rpm
```

Quando a instalação estiver concluída, o Spatial Extender será instalado no diretório /opt/IBM/db2/V8.1 juntamente com outros produtos do DB2.

Depois de instalar o Spatial Extender, crie seu ambiente de instância do DB2 se ainda não o criou e, em seguida, verifique a instalação.

Conceitos Relacionados:

- "Requisitos do Sistema para Instalar o Spatial Extender" na página 26

Tarefas Relacionadas:

- "Installing a DB2 product using rpm (Linux)" na publicação *Suplemento de Instalação e Configuração*
- "Mounting the CD-ROM (Linux)" na publicação *Quick Beginnings for DB2 Servers*
- "Installing a DB2 product using the db2_install script (UNIX)" na publicação *Suplemento de Instalação e Configuração*
- "Verificando a instalação do Spatial Extender" na página 39
- "Resolução de Problemas de Instalação" na página 40

Criando o Ambiente da Instância do DB2 Spatial Extender

Esta tarefa faz parte da principal tarefa de Configurando o Spatial Extender.

Esta seção se aplica somente a plataformas UNIX.

O DB2 Spatial Extender pode ser utilizado com qualquer instância do DB2 criada após a instalação do código do Spatial Extender.

O comando **db2icrt** é utilizado para criar novas instâncias do DB2. Todas as novas instâncias do DB2 criadas após a instalação do DB2 Spatial Extender incluem o DB2 Spatial Extender no ambiente da instância.

As instâncias do DB2 criadas antes da instalação do Spatial Extender não incluem o DB2 Spatial Extender em seus ambientes de instância. Para atualizar instâncias do DB2 existentes, utilize o comando **db2iupdt**. Se estiver utilizando o Centro de Controle do DB2 e tiver criado uma instância para o servidor DB2 Administration antes da instalação do DB2 Spatial Extender, será necessário atualizar esta instância.

Procedimento:

Para atualizar uma instância utilizando o comando **db2iupdt**:

1. Efetue login como um usuário com autoridade root.
2. Execute o seguinte comando:

   ```
   DB2DIR/instance/db2iupdt -a AuthType -u FencedID InstName
   ```

   Em que:

   DB2DIR
   O diretório de instalação do DB2.
   - No AIX, o diretório de instalação do DB2 é /usr/opt/db2_08_01

- Em todos os demais sistemas operacionais baseados em UNIX, o diretório de instalação é /opt/IBM/db2/V8.1

-a AuthType
Representa o tipo de autenticação para a instância. AuthType pode ser um entre SERVER, CLIENT, DCS, SERVER_ENCRYPT, DCS_ENCRYPT. SERVER é o padrão. Este parâmetro é opcional.

-u FencedID
Representa o nome do usuário sob o qual as UDFs (User Defined Functions) limitadas e os procedimentos armazenados limitados serão executados. Este sinalizador não será necessário se você estiver criando uma instância em um cliente do DB2. Especifique o nome do usuário limitado criado.

InstName
Representa o nome da instância. O nome da instância deve ser igual ao nome do usuário proprietário da instância. Especifique o nome do usuário proprietário da instância criada. A instância será criada no diretório pessoal do usuário proprietário da instância.

Para criar uma instância utilizando **db2icrt**:

1. Efetue login como um usuário com autoridade root.
2. Execute o seguinte comando:

```
DB2DIR/instance/db2icrt -a AuthType -u FencedID InstName
```

Em que:

DB2DIR
o diretório de instalação do DB2.

- No AIX, o diretório de instalação do DB2 é /usr/opt/db2_08_01
- Em todos os demais sistemas operacionais baseados em UNIX, o diretório de instalação é /opt/IBM/db2/V8.1

-a AuthType
Representa o tipo de autenticação para a instância. AuthType pode ser um entre SERVER, CLIENT, DCS, SERVER_ENCRYPT, DCS_ENCRYPT. SERVER é o padrão. Este parâmetro é opcional.

-u FencedID
Representa o nome do usuário sob o qual as UDFs (User Defined Functions) limitadas e os procedimentos armazenados limitados serão executados. Este sinalizador não será necessário se você estiver criando uma instância em um cliente do DB2. Especifique o nome do usuário limitado criado.

InstName
Representa o nome da instância. O nome da instância deve ser igual ao nome do usuário proprietário da instância. Especifique o nome do usuário proprietário da instância criada. A instância será criada no diretório pessoal do usuário proprietário da instância.

Por exemplo, se você estiver utilizando autenticação do servidor, o usuário limitado será db2fenc1 e o usuário proprietário da instância será db2inst1. Utilize o seguinte comando para criar uma instância em um sistema AIX:

```
/usr/opt/db2_08_01/instance/db2icrt -a server -u db2fenc1 db2inst1
```

Depois de criar uma instância, você pode querer configurar a notificação para monitoração de funcionamento. Esta tarefa pode ser executada utilizando o Centro de Funcionamento ou o CLP. Consulte o *DB2 Installation and Configuration Supplement* para obter mais informações.

Verificando a instalação do Spatial Extender

Esta tarefa faz parte da tarefa principal de instalando e configurando o Spatial Extender.

Depois de instalar o DB2 Spatial Extender, você pode criar um banco de dados e executar o programa de verificação de instalação para saber se o DB2 Spatial Extender está instalado e configurado corretamente.

Você pode verificar a instalação utilizando o programa de amostra do DB2 Spatial Extender, runGseDemo. O programa runGseDemo foi desenvolvido para resolver problemas com a instalação. Durante a verificação da instalação, você pode receber mensagens de erro que podem ajudá-lo a diagnosticar problemas específicos do sistema. A maioria das mensagens de erro são provocadas por um pequeno número de problemas comuns. Para evitar esses erros, consulte ″Pré-requisitos″.

As etapas de verificação nesta seção se aplicam aos seguintes sistemas operacionais: Windows, AIX, HP-UX, Solaris Operating Environment, Linux para Intel e Linux para S/390.

Pré-requisitos:

Antes de executar o programa runGseDemo:

- Certifique-se de que tenha instalado o produto DB2 Spatial Extender nos ambientes apropriados.
- Utilize um novo banco de dados que não possua nenhuma operação espacial a ele associada.
- Para instalações no UNIX (AIX, HP-UX, Solaris Operating Environment, Linux para Intel e Linux para S/390), verifique se estabeleceu o ambiente de instância do DB2 Spatial Extender. Consulte o *DB2 Installation and Configuration Supplement* para obter informações sobre como executar o programa **db2ilist** para verificar suas instâncias. Pode ser necessário executar o comando **db2start** para iniciar a instância do DB2.
- Aumente o valor do parâmetro de configuração do banco de dados para o tamanho de heap do aplicativo. Para obter detalhes, consulte solucionar problemas da instalação.

Procedimento:

Para verificar a instalação:

1. Somente para UNIX: Efetue logon como o proprietário da instância.
2. Crie um banco de dados.

   Abra a Janela de Comandos do DB2 e insira:

   ```
   db2 create database mydb
   ```

   em que *mydb* é o nome do banco de dados.
3. Localize o programa de verificação de instalação.
   a. Para sistemas operacionais UNIX, insira:

```
cd $HOME/sqllib/samples/spatial
```

em que *$HOME* é o diretório pessoal do proprietário da instância.

b. Para Windows, insira:

```
cd c:\Arquivos de
Programas\IBM\sqllib\samples\spatial
```

em que *c:\Arquivos de Programas\IBM\sqllib* é o diretório no qual você instalou o DB2 Spatial Extender.

4. Execute o programa de verificação da instalação.

Na linha de comando do DB2, insira o comando **runGseDemo**. Por exemplo, insira:

```
runGseDemo mydb userID password
```

em que *mydb* é o nome do banco de dados.

Se você receber mensagens de erro durante o processo de verificação, será necessário solucionar os problemas da instalação.

Tarefas Relacionadas:

- "Resolução de Problemas de Instalação" na página 40
- "Instalando o DB2 Spatial Extender para Windows" na página 27
- "Instalando o DB2 Spatial Extender para AIX" na página 29
- "Instalando o DB2 Spatial Extender para HP-UX" na página 31
- "Instalando o DB2 Spatial Extender para Solaris Operating Environment" na página 33
- "Instalando o DB2 Spatial Extender para Linux" na página 35

Resolução de Problemas de Instalação

Ao executar o programa de amostra (runGseDemo) para verificar se o Spatial Extender foi instalado corretamente, você pode encontrar os seguintes erros:

O banco de dados já está espacialmente ativado

Verifique se o banco de dados para o qual você está verificando a instalação é novo e não possui operações espaciais associadas assim; nesse caso, o programa de amostra falhará.

Você receberá a seguinte mensagem de erro se o banco de dados que está executando o programa de amostra já estiver ativado espacialmente:

```
Ativando o banco de dados logtst...
Returning from ENABLE_DB:
Return code = -14
Return message text =
GSE0014E  O banco de dados já foi ativado para operações espaciais.
```

Para corrigir este problema, elimine o banco de dados e repita as etapas em Verificando a Instalação do Spatial Extender.

Valor do parâmetro de configuração do gerenciador do banco de dados para o tamanho de heap do aplicativo

Se APPLHEAPSZ não estiver definido com um valor adequado, você obterá esta mensagem de erro ao ativar o banco de dados para operações espaciais:

```
GSE0213N  Falha em uma operação de ligação.
SQLERROR = "SQL0001N A ligação ou pré-compilação
não foi concluída com êxito.
SQLSTATE=00000".SQLSTATE=57011
```

Para aumentar o valor do parâmetro de configuração do banco de dados para o tamanho de heap do aplicativo, digite:

```
db2 update db cfg for database_name using APPLHEAPSZ 2048
```

Se 2048 for inadequado, aumente o parâmetro APPLHEAPSZ em incrementos de 256.

Considerações Pós-instalação

Depois de instalar o Spatial Extender, considere o seguinte:

- Fazer download do ArcExplorer
- Acessar dados de referência do geocoder

Fazendo Download do ArcExplorer para DB2

A IBM fornece um navegador, produzido pelo ESRI (Environmental Systems Research Institute) para a IBM, que pode gerar diretamente resultados visuais de consultas de dados do DB2 Spatial Extender sem a necessidade de um servidor de dados intermediário. Esse navegador é o ArcExplorer para DB2. Você pode fazer download de uma cópia gratuita do ArcExplorer para DB2 do Web site do DB2 Spatial Extender da IBM na seguinte localização:

http://www.ibm.com/software/data/spatial/

Para obter mais informações sobre a instalação e o uso do ArcExplorer para DB2, consulte *Using ArcExplorer*, que também está disponível como parte do download do produto ArcExplorer para DB2 no Web site do DB2 Spatial Extender.

Importante: O DB2 Universal Database Versão 8.1 é enviado com o IBM Software Developer's Kit (SDK) para Java Versão 1.3.1. Ao instalar o ArcExplorer para DB2, coloque-o em um diretório separado do DB2. Lembre-se de definir a variável de ambiente CLASSPATH.

Acesso aos Dados de Referência do Geocoder

Os dados de referência do geocoder no CD do DB2 Spatial Extender Geocoder Reference Data foram criados especificamente para funcionar com um geocoder fornecido pelo Spatial Extender. Eles são compostos de dados de mapas base para a rede de ruas dos EUA. O geocoder DB2SE_USA utiliza esses dados para determinar as coordenadas de endereços em um banco de dados ativado para dados espaciais. Estes dados do mapa base são coletivamente chamados de *dados de referência*. O geocoder DB2SE_USA obtém os dados de endereço (não espaciais) em seu banco de dados, compara e os corresponde com os dados de referência e os converte em coordenadas que podem ser armazenadas pelo DB2 Spatial Extender. Esse processo é chamado *geocodificação*.

Os dados de referência fornecidos no CD incluem o arquivo EDGELocator.loc. O arquivo EDGELocator.loc é utilizado pelo geocoder DB2SE_USA para localizar dados de referência específicos. Por exemplo, se você estiver efetuando geocoding de endereços na Califórnia, Kentucky e Oregon, o geocoder DB2SE_USA utilizará o arquivo localizador no CD para determinar as localizações dos endereços.

Procedimento:

Você pode acessar dados do geocoder diretamente do CD, ou pode copiar os dados para sua unidade de disco rígido. Para copiar arquivos de dados do geocoder do CD para o ambiente do servidor DB2 Spatial Extender, execute as etapas explicadas nesta seção.

Para sistemas operacionais UNIX:

1. Monte o CD. Para obter informações sobre como montar um CD, consulte *DB2 for UNIX Quick Beginnings*.
2. Inicie sessão na máquina servidora de destino como um usuário com autoridade de root.
3. Digite um dos seguintes comandos:
   - Para AIX:
     ```
     cp /cdrom/db2/* /usr/opt/db2_08_01/gse/refdata/
     ```
   - Para as demais plataformas UNIX:
     ```
     cp /cdrom/db2/* /0pt/IBM/db2/V8.1/gse/refdata/
     ```

   Nota: Você pode copiar arquivos de dados do geocoder para qualquer diretório em sua unidade local. Se optar por copiar os arquivos para um diretório que você especificar, será necessário modificar o arquivo localizador para apontar para a nova localização.
4. Encerre a sessão.

Para Windows, você pode utilizar a janela Comando ou o Windows Explorer.

Para utilizar a janela de Comando para acessar os dados do geocoder:

1. Clique em **Iniciar** –> **Programa** –> **IBM DB2** –> **Janela de Comando**.
2. Digite o seguinte comando:
   ```
   copy d:\db2\* %db2path%\gse\refdata
   ```

   em que *d:* é a letra que corresponde à sua unidade de CD.

   Nota: Você pode copiar arquivos de dados do geocoder para qualquer diretório em sua unidade local. Se optar por copiar os arquivos para um diretório que você especificar, será necessário modificar o arquivo localizador para apontar para a nova localização.

Para utilizar o Windows Explorer para acessar os dados do geocoder:

Copie todos os arquivos de *d:*\db2 para *c:\Arquivos de Programas\IBM\sqllib*\gse\refdata, em que *d:* é a unidade de CD e *c:\Arquivos de Programas\IBM\sqllib* é o diretório no qual o DB2 está instalado.

Tarefas Relacionadas:

- "Configurando e Instalando o Spatial Extender" na página 25

CDs para Dados e Mapas do DB2 Spatial Extender

O DB2 Spatial Extender é fornecido com sete CDs de dados e de mapas.

As informações de dados e de mapas, identificadas como Dados e Mapas do DB2 Spatial Extender 1 – 7, são fornecidas em sete CDs. A tabela a seguir fornece um

resumo dos dados localizados em cada CD.

Tabela 2. Informações do CD de Dados e de Mapas

CD de Dados e Mapas	Tipo de Resumo de Dados do Mapa
CD 1	Europa e Mundo
CD 2	Canadá, México e Estados Unidos
CD 3	Estados Unidos
CD 4	Estados Unidos (região oeste)
CD 5	Estados Unidos (região central)
CD 6	Estados Unidos (região leste)
CD 7	Estados Unidos (região sul)

Para obter uma descrição detalhada dos dados fornecidos pelo ESRI, consulte o arquivo da ajuda do ESRI, esridata.hlp, localizado no CD Dados e Mapas do DB2 Spatial Extender.

- Para Windows, exiba o arquivo da ajuda em *x:* esridata.hlp, em que *x:* é a unidade de CD.
- Para sistemas operacionais UNIX, exiba ou imprima o arquivo da ajuda localizado no CD em */cdrom/esridata.hlp* , em que */cdrom* é seu ponto de montagem.

Tarefas Relacionadas:

- "Configurando e Ativando o DB2 Geodetic Extender" na página 167

Capítulo 5. Migrando o Ambiente do Spatial Extender para o DB2 Universal Database Versão 8

Esta seção explica como migrar o DB2 Spatial Extender da Versão 8.1 para a Versão 8.2. Também explica como utilizar o utilitário de migração para migrar de um ambiente de 32 bits para um ambiente de 64 bits.

Migrando um Banco de Dados Ativado Espacialmente

Se você utilizou o DB2 Spatial Extender Versão 8.1, deverá concluir as seguintes etapas antes de utilizar um banco de dados existente ativado espacialmente com o DB2 Spatial Extender Versão 8.2 ou o DB2 Geodetic Extender Versão 8.2. Este tópico descreve as etapas requeridas para migrar bancos de dados ativados espacialmente de uma versão anterior do DB2 Spatial Extender.

Pré-requisitos:

Antes de iniciar o processo de migração:

- Encerre todas as conexões com o banco de dados antes de executar o utilitário de migração.
- Certifique-se de que seu sistema atenda os requisitos de instalação para o DB2 Spatial Extender Versão8.2.
- Para fazer backup de um banco de dados, você deve ter autoridade SYSADM, SYSCTRL ou SYSMAINT para o banco de dados.
- Para migrar um banco de dados, você deve ter autoridade SYSADM.

Procedimento:

Para migrar o ambiente do DB2 Spatial Extender:

1. Faça backup do banco de dados da Versão 8.1. Para obter informações sobre como fazer backup de seu banco de dados, consulte o *DB2 Installation and Configuration Supplement*.
2. Instale o DB2 Universal Database Versão 8.2 e o DB2 Spatial Extender Versão 8.2.
3. Migre a instância e os bancos de dados do DB2 da Versão 8.1 para a Versão 8.2. Para obter informações adicionais sobre como migrar a instância e os bancos de dados do DB2, consulte o *DB2 Installation and Configuration Supplement.*
4. Migre um banco de dados ativado espacialmente da Versão 8.1 para a Versão 8.2 utilizando o utilitário de migração do Spatial Extender.
   a. A partir de um prompt de comandos do sistema operacional, utilize o comando **db2se migrate_v82** para migrar o banco de dados.

      ```
      db2se migrate_v82 database_name userId user_id PW password
      ```

      Para obter a sintaxe deste comando, consulte “O Comando db2se migrate_v82” na página 46.

Mensagens de Migração

Se a migração for bem-sucedida, será exibida a seguinte mensagem:

```
GSE0000I A operação foi concluída com êxito
```

Se a migração não for bem-sucedida, será exibida a seguinte mensagem:

```
GSE9002N Ocorreu um erro durante uma tentativa de executar
a migração do banco de dados Spatial Extender.
```

Nota: Os seguintes erros podem ocorrer durante a migração:

- O banco de dados não está ativado espacialmente.
- O banco de dados não é um banco de dados ativado espacialmente Versão 8.1.
- O banco de dados já é um banco de dados ativado espacialmente Versão 8.2.
- O nome do banco de dados não é válido.
- Existem outras conexões com o banco de dados. Executar agora...
- O catálogo espacial não está consistente.
- O usuário não está autorizado.
- A senha não é válida.
- Alguns objetos do usuário não puderam ser migrados.

Verifique o arquivo de mensagens para obter detalhes sobre os erros recebidos. O arquivo de mensagens também contém informações úteis como índices, exibições e a configuração de geocoding que foi migrada.

Tarefas Relacionadas:

- "Backing up databases before DB2 migration" na publicação *Quick Beginnings for DB2 Servers*
- "Migrating databases" na publicação *Quick Beginnings for DB2 Servers*
- "Migrating instances (UNIX)" na publicação *Quick Beginnings for DB2 Servers*
- "Migrating DB2 UDB (Windows)" na publicação *Quick Beginnings for DB2 Servers*
- "Migrating DB2 UDB (UNIX)" na publicação *Quick Beginnings for DB2 Servers*
- "Migrating DB2 Personal Edition (Windows)" na publicação *Quick Beginnings for DB2 Personal Edition*
- "Migrating DB2 Personal Edition (Linux)" na publicação *Quick Beginnings for DB2 Personal Edition*
- "Migrating databases on DB2 Personal Edition (Windows)" na publicação *Quick Beginnings for DB2 Personal Edition*
- "Migrating instances and databases on DB2 Personal Edition (Linux)" na publicação *Quick Beginnings for DB2 Personal Edition*

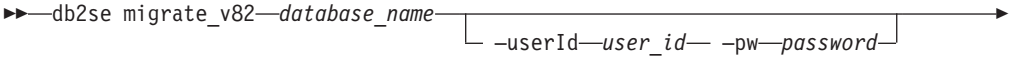
Referência Relacionada:

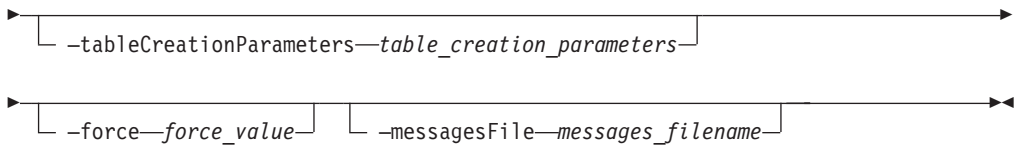
- "O Comando db2se migrate_v82" na página 46

O Comando db2se migrate_v82

Utilize o comando **db2se migrate_v82** para migrar um banco de dados ativado espacialmente da Versão 8.1 para a Versão 8.2.

Sintaxe:





Em que:

-database_name
O nome do banco de dados a ser migrado.

-user_Id
O ID do usuário do banco de dados que tem autoridade SYSADM ou DBADM no banco de dados que está sendo migrado.

-password
Sua senha de usuário.

-messages_filename
O nome do arquivo que contém o relatório de ações de migração. O nome do arquivo fornecido deve ser um nome de arquivo completo no servidor.

Tarefas Relacionadas:

- "Configurando e Ativando o DB2 Geodetic Extender" na página 167
- "Configurando e Instalando o Spatial Extender" na página 25
- "Ativando um Banco de Dados para Operações Espaciais" na página 54

Capítulo 6. Configurando um Banco de Dados

Este capítulo descreve como configurar um banco de dados para acomodar dados espaciais.

Configurando um Banco de Dados para Acomodar Dados Espaciais

O DB2 Spatial Extender, que é executado no ambiente do DB2 Universal Database, funciona com a maioria dos valores de configuração padrão do DB2. No entanto, vários parâmetros de configuração afetam as operações espaciais. Você deve ajustar estes parâmetros para que seus aplicativos espaciais sejam executados com a maior eficiência possível. Em alguns casos, é necessário escolher um valor diferente do valor padrão para operações espaciais. Em outros casos, isso é recomendado dependendo de seus aplicativos e de todo o ambiente do DB2. Este tópico identifica os parâmetros de configuração do DB2 que influenciam as operações do DB2 Spatial Extender.

As seções a seguir explicam como ajustar o gerenciador do banco de dados DB2 e os parâmetros de configuração do banco de dados que afetam as operações do DB2 Spatial Extender.

Ajustando os Parâmetros de Configuração do Banco de Dados

Vários parâmetros de configuração do banco de dados afetam aplicativos espaciais. Para modificar qualquer parâmetro de configuração do banco de dados, você deve estar conectado ao banco de dados. Ao modificar os valores destes parâmetros para um banco de dados, a alteração afetará somente esse banco de dados. As seções a seguir explicam como ajustar os parâmetros para aplicativos espaciais:

- "Ajustando Características do Log de Transação"
- "Ajustando o Tamanho de Heap do Aplicativo" na página 51
- "Ajustando o Tamanho de Heap de Controle do Aplicativo" na página 51

Ajustando Características do Log de Transação

Antes de ativar um banco de dados para operações espaciais, certifique-se de que tenha capacidade suficiente para o log de transação. Os valores padrão para os parâmetros de configuração do log de transação não fornecem capacidade suficiente para o log de transação se você estiver planejando:

- Ativar um banco de dados para operações espaciais em um ambiente Windows
- Utilizar o procedimento armazenado ST_import_shape para importar a partir de arquivos de formas
- Utilizar geocoding com um escopo de consolidação maior
- Executar transações simultâneas

Se seus planos incluírem qualquer uma destas utilizações agora ou no futuro, será necessário aumentar a capacidade do log de transação para o banco de dados, aumentando um ou mais dos parâmetros de configuração do log de transação. Caso contrário, você poderá utilizar as características padrão. Neste caso, prossiga para "Ajustando o Tamanho de Heap do Aplicativo" na página 51.

Recomendação: Consulte a tabela a seguir para obter os valores mínimos recomendados para os três parâmetros de configuração do log de transação.

Tabela 3. Valores Mínimos Recomendados para os Parâmetros de Configuração de Transação

Parâmetro	Descrição	Valor Padrão	Valor Mínimo Recomendado
LOGFILSZ	Especifica o tamanho do arquivo de log como um número de blocos de 4 KB	1000	1000
LOGPRIMARY	Especifica quantos arquivos de log principais devem ser pré-alocados para os arquivos de log de recuperação	3	10
LOGSECOND	Especifica o número de arquivos de log secundários	2	2

Se a capacidade de seu log de transação for inadequada, a seguinte mensagem de erro será emitida quando você tentar ativar um banco de dados para operações espaciais:

```
GSE0010N Não há espaço suficiente no log para o DB2.
```

Procedimento:

Para aumentar o valor de um ou mais parâmetros de configuração:

1. Localize o valor atual dos parâmetros LOGFILSIZ, LOGPRIMARY e LOGSECOND revendo a saída a partir do comando GET DATABASE CONFIGURATION ou da janela **Configurar Banco de Dados** do Centro de Controle do DB2.
2. Decida se deseja alterar um, dois ou três dos valores, conforme indicado na tabela acima.
3. Altere cada valor que deseja modificar. Você pode alterar os valores emitindo um ou mais dos seguintes comandos, em que *db_name* identifica seu banco de dados:

   ```
   UPDATE DATABASE CONFIGURATION FOR db_name USING LOGFILSZ 1000

   UPDATE DATABASE CONFIGURATION FOR db_name USING LOGPRIMARY 10

   UPDATE DATABASE CONFIGURATION FOR db_name USING LOGSECOND 2
   ```

   Se o único parâmetro que você alterar for LOGSECOND, a alteração é efetivada imediatamente. Neste caso, prossiga para "Ajustando o Tamanho de Heap do Aplicativo" na página 51.
4. Se você alterar o parâmetro LOGFILSIZ ou LOGPRIMARY, ou ambos:
   a. Desconecte todos os aplicativos do banco de dados.
   b. Se o banco de dados foi explicitamente ativado, desative-o.

   As alterações nos parâmetros LOGFILSIZ ou LOGPRIMARY ou em ambos, serão efetivadas na próxima vez em que o banco de dados for ativado ou uma conexão com o banco de dados for estabelecida.

Ajustando o Tamanho de Heap do Aplicativo

Utilize o parâmetro de configuração do banco de dados APPLHEAPSZ para especificar o tamanho de heap do aplicativo (em número de páginas de 4 KB). Este parâmetro define o número de páginas de memória privada que estão disponíveis para utilização pelo gerenciador do banco de dados, em nome de um agente ou subagente específico. O heap é alocado quando um agente ou subagente é inicializado para um aplicativo. A quantidade alocada é a quantidade mínima necessária para processar o pedido para o agente ou subagente. Como o agente ou o subagente requer mais espaço no heap para processar instruções SQL maiores, o gerenciador do banco de dados aloca memória conforme necessário, até o máximo especificado neste parâmetro. O heap do aplicativo é alocado fora da memória privada do agente.

O valor padrão para o parâmetro APPLHEAPSZ é 128 (páginas de 4 KB). Ao executar o procedimento armazenado ST_enable_db, este valor deve ser de, pelo menos, 2048.

Recomendação: Para a maioria dos aplicativos do DB2 Spatial Extender, principalmente os que importam ou exportam arquivos de formas, utilize um valor de parâmetro APPLHEAPSZ de pelo menos 2048.

Se o APPLHEAPSZ for definido para um valor inadequado, a seguinte mensagem de erro será emitida quando você tentar ativar um banco de dados para operações espaciais:

```
GSE0009N Não há espaço suficiente disponível no heap de aplicativo do DB2.

GSE0213N Uma operação de ligação falhou. SQLERROR = "SQL0001N A ligação ou
pré-compilação não foi concluída com êxito. SQLSTATE=00000".
```

Procedimento:

Para alterar o tamanho de heap do aplicativo:

1. Localize o valor atual do parâmetro APPLHEAPSZ revendo a saída a partir do comando GET DATABASE CONFIGURATION ou da janela **Configurar Banco de Dados** do Centro de Controle do DB2.
2. Altere o valor para o valor recomendado de 2048 ou para um valor maior. Você pode alterar o valor para 2048 emitindo o seguinte comando, em que *db_name* identifica seu banco de dados:

   ```
   UPDATE DATABASE CONFIGURATION FOR db_name USING APPLHEAPSZ 2048
   ```
3. Desconecte todos os aplicativos do banco de dados.
4. Se o banco de dados foi explicitamente ativado, desative-o.

A alteração será efetivada na próxima vez em que o banco de dados for ativado ou uma conexão com o banco de dados for estabelecida.

Ajustando o Tamanho de Heap de Controle do Aplicativo

Todos os aplicativos do DB2 Spatial Extender, principalmente os que importam ou exportam arquivos de formas, podem se beneficiar da utilização do valor recomendado para o tamanho de heap de controle do aplicativo. Você especifica esta característica com o parâmetro APP_CTL_HEAP_SZ. Este parâmetro especifica o tamanho máximo, em páginas de 4 KB, para a memória compartilhada de controle do aplicativo. Os heaps de controle do aplicativo são alocados a partir desta memória compartilhada. Um heap de controle do aplicativo é alocado para cada aplicativo no banco de dados em que o aplicativo está ativo (ou, no caso de um sistema de banco de dados particionado, em cada partição do banco de dados

em que o aplicativo está ativo). O heap é alocado durante o processamento da conexão pelo primeiro agente que recebe um pedido para o aplicativo no banco de dados (ou na partição do banco de dados). O heap é utilizado para compartilhar informações entre agentes que trabalham em nome do mesmo aplicativo. (Em um ambiente de banco de dados particionado, o compartilhamento ocorre no nível da partição do banco de dados; o compartilhamento não ocorre nas partições do banco de dados.) O valor padrão para o parâmetro APP_CTL_HEAP_SZ é 128.

Recomendação: Para a maioria dos aplicativos do DB2 Spatial Extender, utilize um valor de parâmetro APP_CTL_HEAP_SZ de pelo menos 1024 (páginas de 4 KB).

Se o APP_CTL_HEAP_SZ for definido para um valor inadequado, a seguinte mensagem de erro será emitida quando você importar dados para um banco de dados a partir de arquivos de formas:

```
GSE0214N Uma instrução INSERT falhou. SQLERROR = "SQL0973N
Não há memória suficiente disponível no heap "APP_CTL_HEAP" para processar
a instrução.
```

Procedimento:

Para alterar o tamanho de heap de controle do aplicativo:

1. Localize o valor atual do parâmetro APP_CTL_HEAP_SZ revendo a saída a partir do comando GET DATABASE CONFIGURATION ou da janela **Configurar Banco de Dados** do Centro de Controle do DB2.
2. Altere o valor para o valor recomendado de 1024 (páginas de 4 KB) ou para um valor maior. Você pode emitir o seguinte comando, em que *db_name* identifica seu banco de dados:

   ```
   UPDATE DATABASE CONFIGURATION FOR db_name USING APP_CTL_HEAP_SZ 1024
   ```

3. Desconecte todos os aplicativos do banco de dados.
4. Se o banco de dados foi explicitamente ativado, desative-o.

A alteração será efetivada na próxima vez em que o banco de dados for ativado ou uma conexão com o banco de dados for estabelecida.

Capítulo 7. Configurando Recursos Espaciais para um Banco de Dados

Depois de configurar o banco de dados para adaptar os dados espaciais, você está pronto para fornecer ao banco de dados os recursos que serão necessários ao criar e gerenciar colunas espaciais e analisar dados espaciais. Estes recursos incluem:

- Objetos fornecidos pelo Spatial Extender para suportar operações espaciais; por exemplo, procedimentos armazenados para administrar um banco de dados, tipos de dados espaciais e utilitários espaciais para geocoding e importação ou exportação de dados espaciais.
- Dados de referência: Intervalos de endereços que o DB2SE_USA_GEOCODER utiliza para converter endereços individuais em coordenadas.
- Geocoders fornecidos por usuários ou fornecedores.

Este capítulo descreve estes recursos e apresenta as tarefas através das quais você os torna disponíveis: ativando seu banco de dados para operações espaciais, configurando o acesso para dados de referência e registrando geocoders não-padrão.

Como Configurar Recursos no Banco de Dados

A primeira tarefa que você executa depois de configurar o banco de dados para adaptar os dados espaciais é tornar o banco de dados capaz de suportar operações espaciais—operações, tais como, preencher tabelas com dados espaciais e processar consultas espaciais. Esta tarefa inclui carregar o banco de dados com determinados recursos fornecidos pelo DB2 Spatial Extender. Esta seção descreve esses recursos e destaca a tarefa.

Inventário de Recursos Fornecidos para o Seu Banco de Dados

Para ativar um banco de dados para suportar operações espaciais, o DB2® Spatial Extender fornece o banco de dados com os seguintes recursos:

- Procedimentos armazenados. Ao solicitar uma operação espacial — por exemplo, quando emitir um comando para importar dados espaciais — o DB2 Spatial Extender chamará um destes procedimentos armazenados para executar a operação.
- Tipos de dados espaciais. Você deve atribuir um tipo de dados espaciais a cada coluna da tabela ou exibição que deve conter os dados espaciais.
- Catálogo do DB2 Spatial Extender. Algumas operações dependem deste catálogo. Por exemplo, antes de acessar uma coluna espacial a partir das ferramentas de visualização, a ferramenta pode requerer que a coluna espacial esteja registrada no catálogo.
- Um índice de grade espacial. Permite definir índices de grade em colunas espaciais.
- Funções espaciais. Você as utiliza para trabalhar com dados espaciais de diversas formas; por exemplo, para determinar relacionamentos entre geometrias e para gerar mais dados espaciais.
- Definições de sistemas de coordenadas.

- Sistemas de referência espacial padrão.
- Dois esquemas: DB2GSE e ST_INFORMTN_SCHEMA. O DB2GSE contém os objetos que estão apenas listados: procedimentos armazenados, tipos de dados espaciais, o catálogo do DB2 Spatial Extender e outros. As exibições do catálogo também estão disponíveis no ST_INFORMTN_SCHEMA para estar de acordo com o padrão SQL/MM.

Ativando um Banco de Dados para Operações Espaciais

A tarefa de fazer o DB2 Spatial Extender fornecer um banco de dados com os recursos para criar colunas espaciais e manipular dados espaciais, geralmente é referida como "ativação do banco de dados para operações espaciais".

Pré-requisito:

Antes de ativar um banco de dados para operações espaciais, seu ID de usuário deve ter autoridade SYSADM ou DBADM no banco de dados.

Restrições:

Somente os tipos de dados criados pelo comando enable_db podem ser utilizados.

Procedimento:

Você pode ativar um banco de dados para operações espaciais em qualquer uma das seguintes formas:

- Utilize a janela Ativar Banco de Dados a partir da opção de menu do DB2 Spatial Extender. A opção de menu está disponível a partir do objeto do banco de dados do Centro de Controle do DB2.
- Emita o comando **db2se enable_db**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.ST_enable_db.

Nota:

> Você pode escolher explicitamente o espaço de tabelas no qual deseja que o catálogo do DB2 Spatial Extender resida. Se não escolher, o DB2 escolherá um espaço de tabelas para você.

Conceitos Relacionados:

- "Inventário de Recursos Fornecidos para o Seu Banco de Dados" na página 53

Tarefas Relacionadas:

- "Gerando Aplicativos para o DB2 Spatial Extender" na página 135

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127
- "ST_enable_db" na página 254

Como Trabalhar com Dados de Referência

Esta seção explica o que são dados de referência e declara o que é preciso fazer para acessá-los.

Dados de referência

Os dados de referência são um intervalo de endereços que o DB2SE_USA_GEOCODER utiliza para converter endereços individuais em coordenadas. Estes dados consistem em intervalos dos endereços mais recentes coletados pela United States Census Bureau. Quando DB2SE_USA_GEOCODER lê um endereço no banco de dados, ele procura nos dados de referência:

- Nomes de determinadas ruas na área designada pelo CEP do endereço. O geocoder procura nomes que correspondam o nome da rua no endereço a um grau especificado, ou a um grau maior do que o especificado; por exemplo, 80 por cento ou mais.
- O intervalo de endereços que corresponde ao número do endereço.

Se uma correspondência for encontrada e não tiver o score solicitado, o geocoder retornará as coordenadas do endereço lido. Se não for encontrada uma correspondência ou não tiver o score solicitado, o geocoder retornará nulo.

Um arquivo de configuração avançada chamado *arquivo localizador* pode ser utilizado para influenciar ainda mais o processamento executado pelo geocoder, DB2SE_USA_GEOCODER. A configuração padrão fornecida pelo DB2® Spatial Extender geralmente não precisa ser alterada neste arquivo.

Configurando o Acesso a Dados de Referência

Os dados de referência para o DB2SE_USA_GEOCODER estão em um dos CDs nos quais o Spatial Extender é fornecido. Esta seção descreve como preparar-se para acessá-los.

Procedimento:

Para se preparar para acessar os dados de referência do geocoder padrão:

1. Decida se deseja manter os dados de referência no CD ou armazená-los em sua unidade de disco rígido. Se você os mantiver no CD, economizará espaço (aproximadamente 700 megabytes) que seriam ocupados na unidade de disco rígido. Se você armazená-los na unidade de disco rígido, poderá recuperá-los mais rapidamente do que a partir do CD.
2. Se desejar armazenar os dados de referência em sua unidade de disco rígido:
   a. Verifique se a unidade de disco rígido tem espaço suficiente para conter os dados (aproximadamente 700 megabytes).
   b. Copie os dados para a unidade de disco rígido. Para obter instruções, consulte o README que acompanha os dados de referência.
   c. Verifique se a cópia foi bem-sucedida: Para verificar no UNIX se os dados foram carregados corretamente, procure-os no diretório $DB2INSTANCE/sqllib/gse/refdata/. Para verificar no Windows NT se os dados foram carregados adequadamente, procure-os no diretório %DB2PATH%\gse\refdata\.
3. Informe ao DB2SE_USA_GEOCODER o nome e a localização do arquivo localizador e do mapa base. Faça isso definindo os parâmetros base_map e locator_file do DB2SE_USA_GEOCODER com os valores apropriados. Para obter mais informações, consulte o administrador do banco de dados ou entre em contato com seu representante IBM.

Registrando um Geocoder

O DB2SE_USA_GEOCODER é registrado no DB2 Spatial Extender automaticamente quando um banco de dados é ativado para operações espaciais. Antes que outros geocoders sejam utilizados, eles também devem ser registrados.

Pré-requisito:

Antes de registrar um geocoder, seu ID de usuário deve conter autoridade SYSADM ou DBADM no banco de dados no qual o geocoder reside.

Procedimento:

Você pode registrar um geocoder de uma das seguintes formas:

- Registre-o a partir da janela Registrar Geocoder do Centro de Controle do DB2.
- Emita o comando **db2se register_gc**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.ST_register_geocoder.

Conceitos Relacionados:

- "Geocoders e Geocoding" na página 93

Parte 3. Criando Projetos que Utilizam Dados Espaciais

Capítulo 8. Configurando Recursos Espaciais para um Projeto

Depois que o banco de dados é ativado para operações espaciais, você está pronto para criar projetos que utilizam dados espaciais. Entre os recursos requeridos para cada projeto estão um sistema de coordenadas seguido pelos dados e um sistema de referência espacial que define a extensão da área geográfica que é referenciada pelos dados. Este capítulo:

- Descreve a natureza dos sistemas de coordenadas e informa como criá-los
- Explica o que são sistemas de referência espacial e informa como criá-los

Como Utilizar Sistemas de Coordenadas

Ao planejar um projeto que utiliza dados espaciais, você precisa determinar se os dados devem ser baseados em um dos sistemas de coordenadas que são registrados no catálogo do Spatial Extender. Se nenhum desses sistemas de coordenadas atender aos requisitos, você poderá criar um que atenda aos requisitos. Esta discussão explica o conceito de sistemas de coordenadas e apresenta as tarefas para selecionar um para ser utilizado e criar um novo.

Sistemas de Coordenadas

Um sistema de coordenadas é uma estrutura para definir as localizações relativas de elementos em uma determinada área; por exemplo, uma área da superfície terrestre ou a superfície terrestre como um todo. O DB2® Spatial Extender suporta os seguintes tipos de sistemas de coordenadas para determinar a localização de um recurso geográfico:

Sistema de Coordenadas Geográficas
Um *sistema de coordenadas geográficas* é um sistema de referência (consulte "Sistemas de Referência Espacial" na página 67) que utiliza uma superfície esférica tridimensional para determinar localizações na terra. Qualquer localização na Terra pode ser referida por um ponto com coordenadas de latitude e longitude baseadas em unidades de medida angulares.

Sistema de Coordenadas Projetadas
Um *sistema de coordenadas projetadas* é uma representação plana, bidimensional da Terra. Ele utiliza coordenadas retilíneas (Cartesianas) baseadas em unidades de medida lineares. É baseado em um modelo terrestre esférico (ou esferoidal) e suas coordenadas estão relacionadas a coordenadas geográficas por uma transformação de projeção.

Conceitos Relacionados:

- "Sistema de Coordenadas Geográficas" na página 59
- "Sistema de Coordenadas Projetadas" na página 64

Referência Relacionada:

- "Sistemas de Coordenadas Suportados" na página 529

Sistema de Coordenadas Geográficas

Um *sistema de coordenadas geográficas* é um sistema de referência (consulte "Sistemas de Referência Espacial" na página 67) que utiliza uma superfície esférica tridimensional para determinar localizações na terra. Qualquer localização na terra

pode ser referida por um ponto com coordenadas de longitude e latitude. Os valores para os pontos podem ter as seguintes unidades de medida:

- Unidades lineares quando o sistema de coordenadas geográficas tiver um SRID (Spatial Reference System Identifier) que o DB2® Geodetic Extender reconhece.
- Qualquer uma das seguintes unidades quando o sistema de coordenadas geográficas tiver um SRID que o DB2 Geodetic Extender não reconhece.
  - Graus decimais
  - Minutos decimais
  - Segundos decimais
  - Gradianos
  - Radianos

Para obter o intervalo de valores para estas unidades, consulte "Sistemas de Coordenadas Suportados" na página 529.

Por exemplo, a Figura 6 mostra um sistema de coordenadas geográficas no qual uma localização é representada pelas coordenadas de 80 graus de longitude ao Leste e 55 graus de latitude ao Norte.

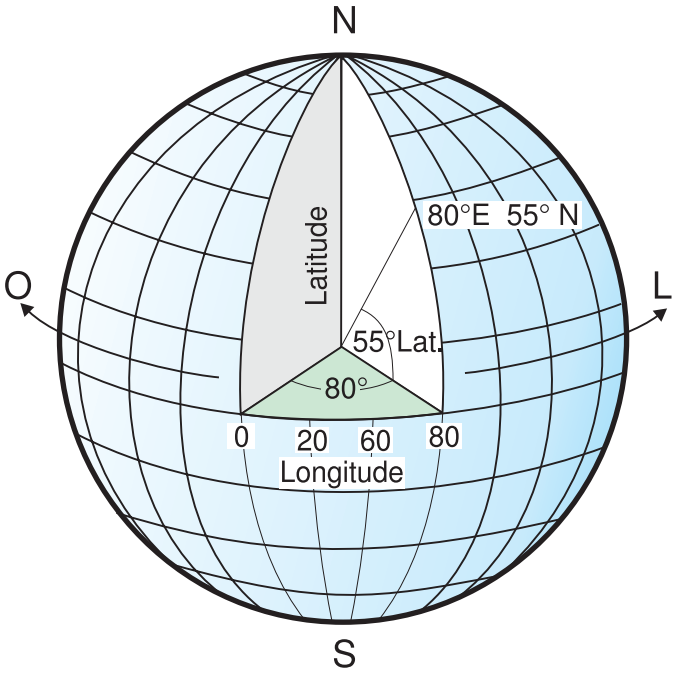


Figura 6. Um Sistema de Coordenadas Geográficas

Cada uma das linhas que percorrem o leste e o oeste possui um valor de latitude constante e são chamadas de *paralelos*. Elas são eqüidistantes e paralelas e formam círculos concêntricos ao redor da terra. O *equador* é o maior círculo e divide a terra ao meio. Tem a mesma distância a partir de cada pólo e o valor desta linha latitudinal é zero. As localizações ao norte do equador possuem latitudes positivas que vão de 0 a +90 graus, enquanto as localizações ao sul do equador possuem latitudes negativas que vão de 0 a -90 graus.

A Figura 7 na página 61 ilustra linhas latitudinais.

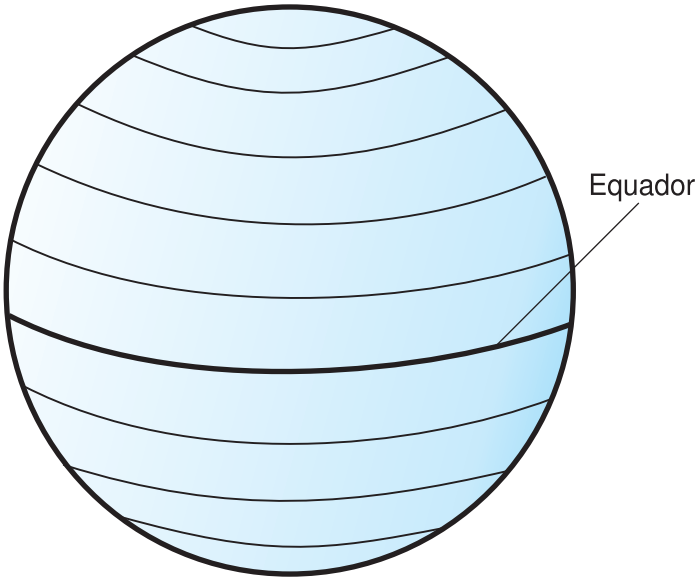


Figura 7. Linhas Latitudinais

Cada uma das linhas que percorrem o norte e o sul possuem um valor de longitude constante e são chamadas de *meridianos*. Elas formam círculos do mesmo tamanho ao redor da terra e se cruzam nos pólos. A *linha do meridiano* é a linha longitudinal que define a origem (zero graus) para coordenadas longitudinais. Uma das localizações mais utilizadas da linha do meridiano é a linha que passa por Greenwich, Inglaterra. No entanto, outras linhas longitudinais, como as que passam por Berna, Bogotá e Paris, também foram utilizadas como o meridiano principal. As localizações a leste do meridiano principal até seu meridiano *antipodal* (a continuação do meridiano principal no outro lado do globo) possuem longitudes positivas que vão de 0 a +180 graus. As localizações a oeste do meridiano principal possuem longitudes negativas que vão de 0 a –180 graus.

A Figura 8 ilustra linhas longitudinais.

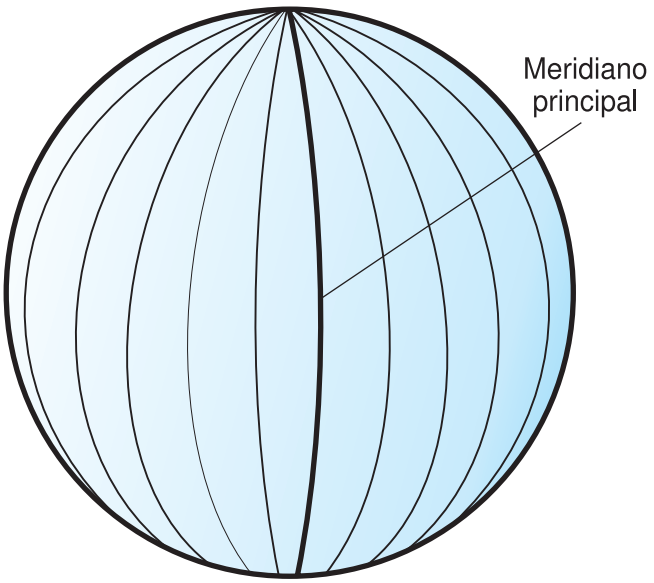


Figura 8. Linhas Longitudinais

As linhas latitudinais e longitudinais podem cobrir o globo para formar uma grade, chamada de *gratícula*. O ponto de origem da gratícula é (0,0), em que a linha do equador e a linha do meridiano se cruzam. O equador é o único lugar na gratícula em que a distância linear correspondente à latitude de um grau é aproximadamente igual à distância correspondente à longitude de um grau. Como as linhas longitudinais convergem nos pólos, a distância entre dois meridianos é diferente em cada paralelo. Portanto, conforme você se aproxima dos pólos, a distância correspondente à latitude de um grau será muito maior do que a correspondente à longitude de um grau.

Também é difícil determinar as profundidades das linhas latitudinais utilizando a gratícula. As linhas latitudinais são círculos concêntricos que ficam menores próximos aos pólos. Elas formam um único ponto nos pólos nos quais os meridianos começam. No equador, um grau de longitude equivale a aproximadamente 111.321 quilômetros, enquanto a 60 graus de latitude, um grau de longitude equivale a apenas 55.802 km (esta aproximação é baseada no esferóide Clarke 1866). Portanto, como não existe nenhum comprimento uniforme de graus de latitude e longitude, a distância entre pontos não pode ser calculada com precisão utilizando unidades de medida angulares.

A Figura 9 mostra as diferentes dimensões entre localizações na gratícula.

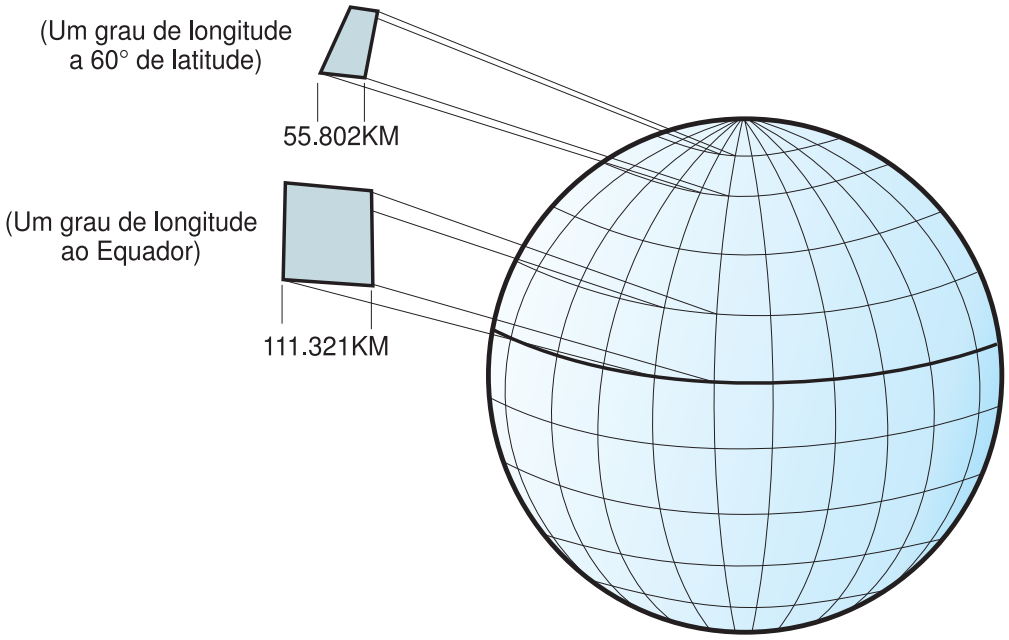


Figura 9. Diferentes Dimensões entre Localizações na Gratícula

Um sistema de coordenadas pode ser definido por uma aproximação de esfera ou esferóide do formato da terra. Como a Terra não é perfeitamente redonda, um esferóide pode ajudar a manter a precisão de um mapa, dependendo da localização na Terra. Um *esferóide* é um elipsóide baseado em uma elipse, enquanto uma esfera é baseada em um círculo.

O formato da elipse é determinado por dois raios. O raio mais longo é chamado de semi-eixo maior e o raio mais curto é chamado de semi-eixo menor. Um elipsóide é um formato tridimensional formado pela rotação de uma elipse em torno de um de seus eixos.

A Figura 10 mostra as aproximações de esfera e de esferóide da Terra e os eixos maior e menor de uma elipse.

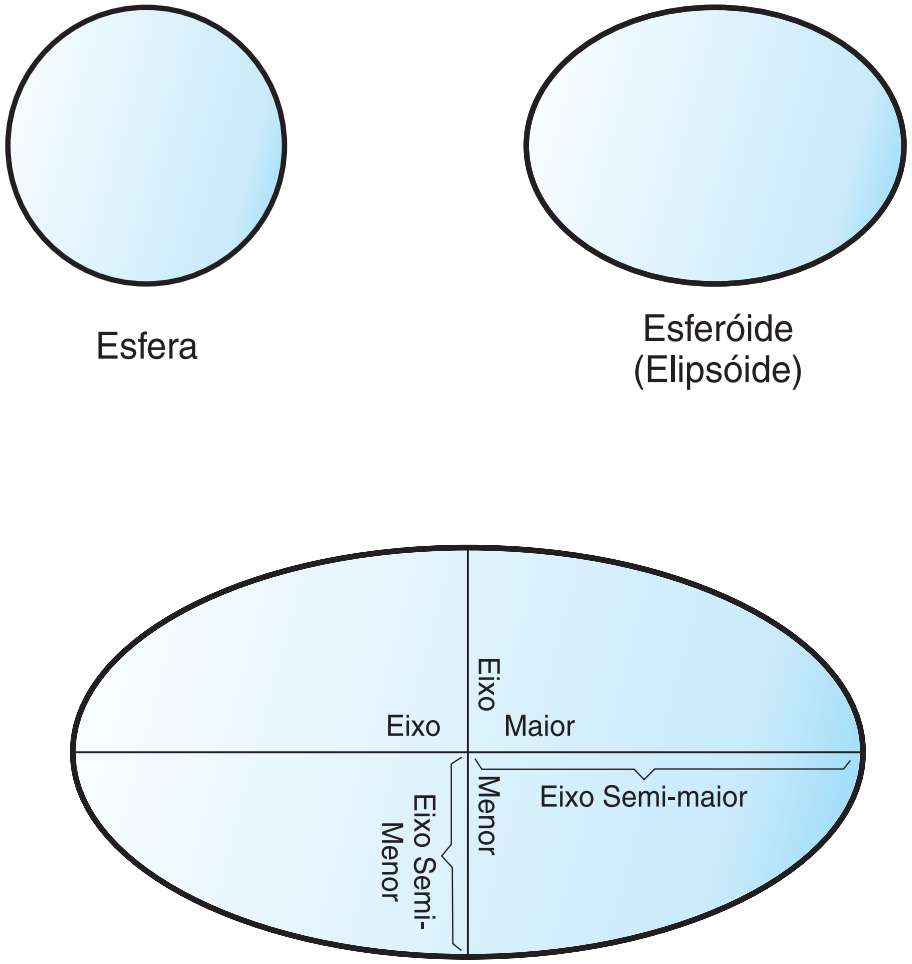


Figura 10. Aproximações de Esfera e de Esferóide

Um *datum* é um conjunto de valores que definem a posição do esferóide relativo ao centro da terra. O datum fornece um quadro de referência para medir localizações e define a origem e a orientação de linhas latitudinais e longitudinais. Alguns datums são globais e destinam-se a fornecer uma boa média de precisão ao redor do mundo. Um datum local alinha seu esferóide para ajustar a superfície da terra em uma determinada área. Portanto, as medidas do sistema de coordenadas não serão precisas se forem utilizadas com uma área diferente da área para a qual foram designadas. Para obter informações adicionais sobre elipsóides, consulte "Sistemas de Coordenadas Suportados" na página 529.

A Figura 11 na página 64 mostra as diferenças de alinhamento de datums com a superfície da terra. O datum local, NAD27, está mais alinhado com a superfície da Terra do que o datum centralizado na Terra, WGS84, nesta localização específica.

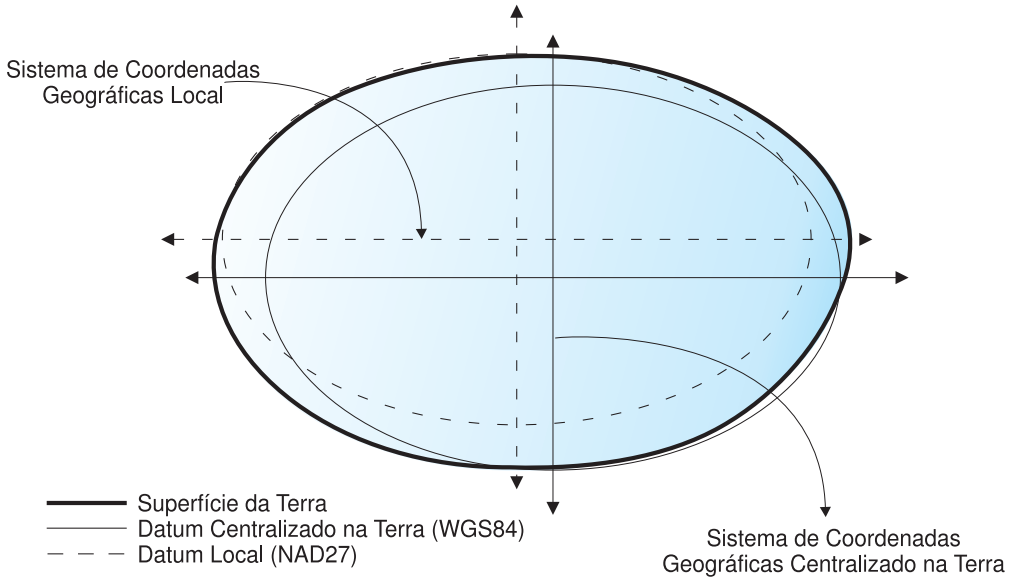


Figura 11. Alinhamentos de Datums

Sempre que você alterar o dado, o sistema de coordenadas geográficas é alterado e os valores de coordenadas serão alterados. Por exemplo, as coordenadas em DMS de um ponto de controle em Redlands, Califórnia, que utilizam o North American Datum de 1983 (NAD 1983) são: "-117 12 57.75961 34 01 43.77884" As coordenadas do mesmo ponto no North American Datum de 1927 (NAD 1927) são: "-117 12 54.61539 34 01 43.72995".

Sistema de Coordenadas Projetadas

Um *sistema de coordenadas projetadas* é uma representação plana, bidimensional da Terra. Ele é baseado em um sistema de coordenadas geográficas de esfera ou esferóide, mas utiliza unidades de medida lineares para coordenadas, para que os cálculos de distância e área sejam feitos facilmente com relação a estas mesmas unidades.

As coordenadas latitudinais e longitudinais são convertidas em coordenadas x, y na projeção plana. A coordenada x geralmente representa a direção leste de um ponto e a coordenada y geralmente representa a direção norte de um ponto. A linha central que percorre o leste e o oeste é referida como o eixo de x e a linha central que percorre o norte e o sul é referida como o eixo de y.

A interseção dos eixos de x e de y é a origem e geralmente possui uma coordenada (0,0). Os valores acima do eixo x são positivos e os valores abaixo do eixo x são negativos. As linhas paralelas ao eixo de x são eqüidistantes umas das outras. Os valores à direita do eixo y são positivos e os valores à esquerda do eixo y são negativos. As linhas paralelas ao eixo de y são eqüidistantes.

São utilizadas fórmulas matemáticas para converter um sistema de coordenadas geográficas tridimensional em um sistema de coordenadas projetado plano bidimensional. A transformação é referida como uma *projeção de mapa*. As projeções de mapas, geralmente, são classificadas pela superfície da projeção utilizada, como superfícies cônicas, cilíndricas e planas. Dependendo da projeção utilizada, as propriedades espaciais diferentes aparecerão distorcidas. As projeções são designadas para reduzir a distorção de uma ou duas características de dados,

ainda que a distância, área, forma, direção ou uma combinação destas propriedades podem não ser representações precisas dos dados que estão sendo modelados. Existem vários tipos de projeções disponíveis. Enquanto a maioria das projeções de mapas tentam preservar alguma precisão das propriedades espaciais, existem outras que tentam reduzir a distorção geral, como a projeção de *Robinson*. Os tipos mais comuns de projeções de mapa incluem:

Projeções de áreas iguais
Estas projeções preservam a área de recursos específicos. Estas projeções distorcem a forma, o ângulo e a escala. A projeção *Albers Equal Area Conic* é um exemplo de uma projeção de áreas iguais.

Projeções conformais
Estas projeções preservam a forma local para pequenas áreas. Estas projeções preservam ângulos individuais para descrever relacionamentos espaciais mostrando linhas de gratículas perpendiculares que se cruzam em ângulos de 90 graus no mapa. Todos os ângulos são preservados; porém, a área do mapa é distorcida. As projeções *Mercator* e *Lambert Conformal Conic* são exemplos de projeções conformais.

Projeções eqüidistantes
Estas projeções preservam as distâncias entre determinados pontos, mantendo a escala de um determinado conjunto de dados. Algumas das distâncias podem ser distâncias reais, que são as mesmas distâncias na mesma escala do globo. Se você sair do conjunto de dados, a escala ficará mais distorcida. A projeção *Sinuosa* e a projeção *Cônica eqüidistante* são exemplos de projeções eqüidistantes.

Projeções de direção real ou azimutais
Estas projeções preservam a direção de um ponto para todos os demais pontos, mantendo alguns dos arcos de círculos grandes. Estas projeções fornecem as direções ou ângulos de fundo de todos os pontos no mapa corretamente com relação ao centro. Os mapas azimutais podem ser combinados com projeções de áreas iguais, conformais e eqüidistantes. A projeção *Lambert Equal Area Azimutal* e a projeção *Azimutal Eqüidistante* são exemplos de projeções azimutais.

Conceitos Relacionados:

- "Sistema de Coordenadas Geográficas" na página 59
- "Sistemas de Coordenadas" na página 59

Tarefas Relacionadas:

- "Gerando Aplicativos para o DB2 Spatial Extender" na página 135

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127
- "ST_create_coordsys" na página 238
- "Sistemas de Coordenadas Suportados" na página 529

Selecionando ou Criando Sistemas de Coordenadas

Depois de ativar um banco de dados para operações espaciais, você estará pronto para planejar projetos que utilizem dados espaciais. A primeira etapa no planejamento de um projeto é determinar qual sistema de coordenadas utilizar. Suas opções são as seguintes:

- Você pode utilizar um sistema de coordenadas que foi fornecido com o DB2 Spatial Extender ou um que foi criado por um usuário. Mais de 2000 sistemas de coordenadas são fornecidos com o DB2 Spatial Extender. Entre eles estão:
  - Um sistema de coordenadas ao qual o DB2 Spatial Extender se refere como "Não especificado." Utilize este sistema de coordenadas quando:
    - Precisar definir localizações que não têm relacionamento direto com a superfície terrestre; por exemplo, localizações de escritórios em um edifício comercial ou localizações de prateleiras em um armazém.
    - Você pode definir estas localizações em termos de coordenadas positivas que incluem poucos ou nenhum valor decimal.
  - GCS_NORTH_AMERICAN_1983. Utilize este sistema de coordenadas quando precisar definir localizações nos Estados Unidos; por exemplo:
    - Quando importar dados espaciais para os Estados Unidos em CDs de "Mapas e Dados" fornecidos com o DB2 Spatial Extender.
    - Quando planejar utilizar o geocoder fornecido com o DB2 Spatial Extender para endereços de geocode nos Estados Unidos

  Para obter mais informações sobre estes sistemas de coordenadas e para determinar quais outros sistemas de coordenadas foram fornecidos com o DB2 Spatial Extender e quais sistemas de coordenadas (se houver algum) foram criados por outros usuários, consulte a exibição do catálogo DB2SE.ST_COORDINATE_SYSTEMS.
- Você pode criar um sistema de coordenadas.

Pré-requisitos:

Antes de criar um sistema de coordenadas, seu ID de usuário deve ter a autoridade SYSADM ou DBADM no banco de dados que foi ativado para operações espaciais. Não é necessária nenhuma autorização para utilizar um sistema de coordenadas existente.

Procedimento:

Você pode criar um sistema de coordenadas de uma das seguintes formas:

- Crie-o a partir da janela Criar Sistema de Coordenadas do Centro de Controle do DB2.
- Emita o comando **db2se create_cs** a partir do processador da linha de comandos do db2se.
- Execute um aplicativo que chame o procedimento armazenado db2se.ST_create_coordsys.

Conceitos Relacionados:

- "Sistemas de Coordenadas" na página 59

Tarefas Relacionadas:

- "Gerando Aplicativos para o DB2 Spatial Extender" na página 135

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127
- "ST_create_coordsys" na página 238

Como Configurar Sistemas de Referência Espacial

Ao planejar um projeto que utiliza dados espaciais, você precisa determinar se qualquer um dos sistemas de referência espacial disponíveis pode ser utilizado para esses dados. Se nenhum dos sistemas disponíveis for apropriado para os dados, você poderá criar um apropriado. Esta seção explica o conceito de sistemas de referência espacial e descreve as tarefas para selecionar um para ser utilizado e criar um novo.

Sistemas de Referência Espacial

Um *sistema de referência espacial* é um conjunto de parâmetros que inclui:

- O nome do sistema de coordenadas a partir do qual as coordenadas são derivadas.
- O identificador numérico que identifica exclusivamente o sistema de referência espacial.
- Coordenadas que definem a máxima extensão possível de espaço referido por um determinado intervalo de coordenadas.
- Números que, quando aplicados em certas operações matemáticas, convertem coordenadas recebidas como entrada em valores que podem ser processados com eficiência máxima.

As seções a seguir discutem os valores de parâmetros que definem um identificador, uma extensão máxima de espaço e fatores de conversão.

Spatial Reference System Identifier:

O SRID (Spatial Reference System Identifier) é utilizado como um parâmetro de entrada para diversas funções espaciais.

Para um sistema de referência espacial geodésico, o valor do SRID deve estar no intervalo de 2000000000 a 2000001000. O DB2® Geodetic Extender fornece 318 SRS (Spatial Reference Systems) geodésicos. Para obter informações adicionais, consulte "DB2 Geodetic Extender" na página 159.

Definindo o espaço que abrange coordenadas armazenadas em uma coluna espacial:

As coordenadas em uma coluna espacial geralmente definem localizações que se estendem por parte da Terra. O espaço no qual ocorre a extensão — de leste a oeste e de norte a sul — é chamado de *extensão espacial*. Por exemplo, considere um grupo de planícies aluviais cujas coordenadas estão armazenadas em uma coluna espacial. Suponha que mais a oeste e mais a leste destas coordenadas estejam valores latitudinais de –24.556 e –19.338, respectivamente, e que mais ao norte e mais ao sul das coordenadas estejam valores longitudinais de 18.819 e 15.809 graus, respectivamente. A extensão espacial das planícies aluviais é um espaço que se estende por um plano de oeste a leste entre as duas latitudes e um plano de norte a sul entre as duas longitudes. Você pode incluir estes valores em um sistema de referência espacial, atribuindo-os a determinados parâmetros. Se a coluna espacial incluir coordenadas e medidas Z, será necessário incluir também as coordenadas e medidas Z maiores e menores no sistema de referência espacial.

O termo *extensão espacial* pode se referir não somente a uma expansão real de localizações, como no parágrafo anterior; mas também a uma possível expansão. Suponha que as planícies aluviais, no exemplo anterior, tivessem que ampliar pelos

próximos cinco anos. Você poderia estimar quais as coordenadas mais a oeste, mais a leste, mais ao norte e mais ao sul dos planos estaria no final do quinto ano. Você, então, poderia atribuir essas estimativas, em vez das coordenadas atuais, aos parâmetros de uma extensão espacial. Dessa forma, você poderia manter o sistema de referência espacial à medida que as planícies se expandem e suas latitudes e longitudes maiores são incluídas na coluna espacial. Caso contrário, se o sistema de referência espacial estiver limitado às latitudes e longitudes originais, ele precisaria ser alterado ou substituído à medida que as planícies aluviais se expandem.

Convertendo em valores que melhoram o desempenho:

Geralmente, a maioria das coordenadas em um sistema de coordenadas são valores decimais; algumas são inteiros. Além disso, as coordenadas a leste da origem são positivas; as que estão a oeste são negativas. Antes de serem armazenadas pelo Spatial Extender, as coordenadas negativas são convertidas em valores positivos e as coordenadas decimais são convertidas em inteiros. Como resultado, todas as coordenadas são armazenadas pelo Spatial Extender como inteiros positivos. A finalidade é melhorar o desempenho quando as coordenadas são processadas.

Alguns parâmetros em um sistema de referência espacial são utilizados para fazer as conversões descritas no parágrafo precedente. Um parâmetro, chamado de *deslocamento*, é subtraído de cada coordenada negativa, que deixa um valor positivo como restante. Cada coordenada decimal é multiplicada por outro parâmetro, chamado de *fator de escala*, que resulta em um inteiro cuja precisão é igual à da coordenada decimal. (O deslocamento é subtraído de coordenadas positivas e de negativas; e as coordenadas não decimais, bem como as decimais, são multiplicadas pelo fator de escala. Dessa forma, todas as coordenadas positivas e não decimais permanecem correspondentes às decimais.)

Essas conversões ocorrem internamente e permanecem em vigor até que as coordenadas sejam recuperadas. Os resultados de entrada e de consulta sempre contêm coordenadas em seu formato original, não convertido.

Conceitos Relacionados:

- "Fatores de Conversão que Transformam Dados de Coordenadas em Inteiros" na página 73
- "Sistemas de Coordenadas" na página 59

Tarefas Relacionadas:

- "Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema" na página 68
- "Criando um Sistema de Referência Espacial" na página 74

Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema

Depois de determinar qual sistema de coordenadas será utilizado, você estará pronto para fornecer um sistema de referência espacial que seja adequado aos dados de coordenadas com os quais você está trabalhando. O DB2 Spatial Extender fornece cinco sistemas de referência espacial para dados espaciais e o DB2 Geodetic Extender fornece 318 sistemas de referência espacial geodésicos para dados geodésicos.

Procedimento:

Para determinar se você pode utilizar um dos sistemas de referência espacial padrão ou sistemas de referência geodésicos predefinidos:

1. Responda as seguintes perguntas:
   - O sistema de coordenadas no qual o sistema está baseado abrange a área geográfica com a qual você está trabalhando?

     Estes sistemas de coordenadas são mostrados em "Sistemas de Referência Espaciais Fornecidos com o DB2 Spatial Extender" na página 70.
   - Seus dados estão em um sistema de coordenadas geográficas que utiliza Graus ou Graduações Decimais como a unidade de medida? Seus dados se estendem por uma grande parte da superfície da Terra? Você precisa fazer cálculos precisos de distância, comprimento e área? Alguns de seus dados estão próximos do pólo norte, do pólo sul ou do meridiano de data internacional?

     Se você responder sim para qualquer uma destas perguntas, poderá utilizar um dos 318 sistemas de referência espacial geodésicos predefinidos. Para obter informações sobre estes sistemas de referência espacial geodésicos predefinidos, consulte "Datums Suportados pelo DB2 Geodetic Extender" na página 213.
   - Os fatores de conversão associado a um dos sistemas de referência espacial padrão trabalham com dados de coordenadas?

     O Spatial Extender utiliza valores de *deslocamento* e fatores de *escala* para converter dados de coordenadas fornecidos em inteiros positivos. Para determinar se os dados de coordenadas funcionam com os valores de deslocamento e fatores de escala fornecidos para um dos sistemas de referência espacial padrão:

     a. Reveja as informações em "Fatores de Conversão que Transformam Dados de Coordenadas em Inteiros" na página 73.

     b. Veja como estes fatores são definidos para os sistemas de referência espacial padrão. Se, depois de aplicar o valor de deslocamento para as coordenadas mínimas X e Y, elas não forem ambas maiores que 0, você deverá criar um novo sistema de referência espacial e definir os deslocamentos manualmente. Para obter informações adicionais sobre como criar um novo sistema de referência espacial, consulte "Criando um Sistema de Referência Espacial" na página 74.
   - Os dados com os quais você está trabalhando incluem coordenadas de altura e profundidade (coordenadas Z) ou medidas (coordenadas M)?

     Se estiver trabalhando com coordenadas Z ou M, talvez seja necessário criar um novo sistema de referência espacial com deslocamentos Z ou M e fatores de escala adequados para seus dados.
2. Se os sistemas de referência espacial ou sistemas de referência geodésicos existentes não funcionarem com seus dados, será necessário "Criando um Sistema de Referência Espacial" na página 74.

Depois que decidir de qual sistema de referência espacial você precisa, especifique esta opção para o Spatial Extender quando executar uma das seguintes tarefas:

- "Criando Colunas Espaciais" na página 83
- "Registrando Colunas Espaciais" na página 85

Conceitos Relacionados:

- "Fatores de Conversão que Transformam Dados de Coordenadas em Inteiros" na página 73
- "Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender" na página 160
- "Sistemas de Referência Espacial" na página 67

Tarefas Relacionadas:

- "Criando um Sistema de Referência Espacial" na página 74
- "Criando Colunas Espaciais" na página 83
- "Registrando Colunas Espaciais" na página 85
- "Creating a spatial reference system: Spatial Extender help"
- "Registering a spatial column with a spatialreference system: Spatial Extender help"
- "Selecting a spatial reference system : Spatial Extender help"

Referência Relacionada:

- "Sistemas de Referência Espaciais Fornecidos com o DB2 Spatial Extender" na página 70
- "Datums Suportados pelo DB2 Geodetic Extender" na página 213

Sistemas de Referência Espaciais Fornecidos com o DB2 Spatial Extender

O DB2 Spatial Extender fornece os sistemas de referência espaciais que são mostrados na tabela abaixo, junto com o sistema de coordenadas no qual cada sistema de referência espacial é baseado e os valores de deslocamento e fatores de escala que o DB2 Spatial Extender utiliza para converter os dados de coordenadas em inteiros positivos. Você pode localizar informações sobre estes sistemas de referência espaciais na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS.

Se estiver trabalhando com graus decimais (todos os dados nos CDs de dados de amostra do DB2 Spatial Extender estão em graus decimais), os valores de deslocamento e os fatores de escala dos sistemas de referência espacial padrão suportarão o intervalo completo de coordenadas de latitude/longitude e manterão 6 posições decimais, equivalentes a aproximadamente 10 cm.

Se você planeja utilizar o geocoder, que funciona apenas com endereços americanos, selecione ou crie um sistema de referência espacial que trate de coordenadas americanas, como o sistema de coordenadas GCS_NORTH_AMERICAN_1983. Se você não especificar de qual sistema de coordenadas seus dados espaciais devem ser derivados, o Spatial Extender utilizará o sistema de referência espacial DEFAULT_SRS.

Utilize a tabela abaixo para decidir se será utilizado um sistema de referência espacial padrão ou se será criado um novo sistema (consulte "Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema" na página 68. Se nenhum sistema de referência espacial padrão atender suas necessidades, você poderá criar um novo. Consulte "Criando um Sistema de Referência Espacial" na página 74 para obter informações adicionais.

Tabela 4. Sistemas de Referência Espacial Fornecidos com o DB2 Spatial Extender

Sistema de Referência Espacial	ID SRS	Sistema de Coordenadas	Valores de Deslocamento	Fatores de Escala	Quando Utilizar
DEFAULT _SRS	0	Nenhum	xOffset = 0 yOffset = 0 zOffset = 0 mOffset = 0	xScale = 1 yScale = 1 zScale = 1 mScale = 1	Você pode selecionar este sistema quando os dados são independentes de um sistema de coordenadas ou você não pode ou não precisa especificar um.
NAD83_ SRS_1	1	GCS_NORTH _AMERICAN _1983	xOffset = –180 yOffset = –90 zOffset = 0 mOffset = 0	xScale = 1,000,000 yScale = 1,000,000 zScale = 1 mScale = 1	Você poderá selecionar esse sistema de referência espacial se planeja utilizar os dados de amostra americanos fornecidos com o DB2 Spatial Extender. Se os dados de coordenadas com os quais está trabalhando foram coletados após 1983, utilize este sistema em lugar do NAD27_SRS_1002.

Tabela 4. Sistemas de Referência Espacial Fornecidos com o DB2 Spatial Extender (continuação)

Sistema de Referência Espacial	ID SRS	Sistema de Coordenadas	Valores de Deslocamento	Fatores de Escala	Quando Utilizar
NAD27_ SRS_1002	1002	GCS_NORTH _AMERICAN _1927	xOffset = –180 yOffset = –90 zOffset = 0 mOffset = 0	xScale = 5,965,232 yScale = 5,965,232 zScale = 1 mScale = 1	Você poderá selecionar esse sistema de referência espacial se planeja utilizar os dados de amostra americanos fornecidos com o DB2 Spatial Extender. Se os dados de coordenadas com os quais está trabalhando foram coletados após 1983, utilize este sistema em lugar de NAD83_SRS_1. Esse sistema fornece um grau de precisão maior que os outros sistemas de referência espacial padrão.
WGS84_ SRS_1003	1003	GCS_WGS _1984	xOffset = –180 yOffset = –90 zOffset = 0 mOffset = 0	xScale = 5,965,232 yScale = 5,965,232 zScale = 1 mScale = 1	Você poderá selecionar esse sistema de referência espacial se estiver trabalhando com dados fora dos Estados Unidos (Este sistema manipula coordenadas no mundo todo). Não o utilize se planeja usar o geocoder padrão fornecido com o DB2 Spatial Extender, pois ele se destina somente a endereços americanos.
DE_HDN _SRS_1004	1004	GCSW _DEUTSCHE _HAUPTDRE IECKSNETZ	xOffset = –180 yOffset = –90 zOffset = 0 mOffset = 0	xScale = 5,965,232 yScale = 5,965,232 zScale = 1 mScale = 1	Esse sistema de referência espacial baseia-se em um sistema de coordenadas para endereços alemães.

Conceitos Relacionados:

- "Sistemas de Referência Espacial" na página 67

Referência Relacionada:

- "A Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS" na página 295

Fatores de Conversão que Transformam Dados de Coordenadas em Inteiros

O DB2® Spatial Extender utiliza valores de *deslocamento* e fatores de *escala* para converter os dados de coordenadas que você fornece em inteiros positivos. Os sistemas de referência espacial padrão já possuem valor de deslocamento e fatores de escala associados a eles. Se você estiver criando um novo sistema de referência espacial, será necessário determinar fatores de escala e, opcionalmente, os valores de deslocamento que funcionam melhor com seus dados. Para obter informações adicionais, consulte "Criando um Sistema de Referência Espacial" na página 74.

Valores de Deslocamento

Um valor de deslocamento é um número que é subtraído de todas as coordenadas, deixando apenas valores positivos como restante. O Spatial Extender converte seus dados de coordenadas utilizando as seguintes fórmulas para assegurar que todos os valores de coordenadas ajustados sejam maiores que 0.

Notação da fórmula: Nestas fórmulas, a notação "min" representa "o mínimo de todos". Por exemplo, "min(x)" significa "o mínimo de todas as coordenadas x". O deslocamento para cada direção geográfica é representado como o deslocamento da *dimensão*. Por exemplo, xOffset é o valor de deslocamento aplicado a todas as coordenadas X.

$$\min(x) - xOffset \geq 0$$
$$\min(y) - yOffset \geq 0$$
$$\min(z) - zOffset \geq 0$$
$$\min(m) - mOffset \geq 0$$

Fatores de Escala

Um fator de escala é um valor que, quando multiplicado por coordenadas e medidas decimais, resulta em inteiros com pelo menos o mesmo número de dígitos significativos das coordenadas e medidas originais. O Spatial Extender converte seus dados de coordenadas decimais utilizando as seguintes fórmulas para assegurar que todos os valores de coordenadas ajustados sejam inteiros positivos. Os valores convertidos não podem exceder 2^{53} (aproximadamente $9 * 10^{15}$).

Notação da fórmula: Nestas fórmulas, a notação "max" representa "o máximo de todos". O deslocamento de cada dimensão geográfica é representado como um deslocamento de *dimensão* (por exemplo, xOffset é o valor de deslocamento aplicado a todas as coordenadas X). O fator de escala de cada dimensão geográfica é representado como Escala de *dimensão* (por exemplo, xScale é o fator de escala aplicado a coordenadas X).

$$(\max(x) - xOffset) * xScale \leq 2^{53}$$
$$(\max(y) - yOffset) * yScale \leq 2^{53}$$
$$(\max(z) - zOffset) * zScale \leq 2^{53}$$
$$(\max(m) - mOffset) * mScale \leq 2^{53}$$

Ao escolher quais fatores de escala funcionam melhor com os dados de coordenadas, certifique-se de:

- Utilizar o mesmo fator de escala para as coordenadas X e Y.
- Quando multiplicado por uma coordenada X decimal ou uma coordenada Y decimal, o fator de escala gera um valor menor que 2^{53}. Uma técnica comum é tornar o fator de escala uma potência de 10. Ou seja, o fator de escala deve ser 10 para a primeira potência (10), 10 para a segunda potência (100), 10 para a terceira potência (1000), ou, se necessário, um fator maior.
- O fator de escala é grande o suficiente para assegurar que o número de dígitos significativos no novo inteiro seja igual ao número de dígitos significativos na coordenada decimal original.

Exemplo:

Suponha que a função ST_Point receba uma entrada que consiste em uma coordenada X de 10.01, em uma coordenada Y de 20.03 e no identificador de um sistema de referência espacial. Quando ST_Point é chamado, ele multiplica o valor de 10,01 e o valor de 20,03 pelo fator de escala do sistema de referência espacial para as coordenadas X e Y. Se esse fator de escala for 10, os inteiros resultantes que o Spatial Extender armazenará serão 100 e 200, respectivamente. Como o número de dígitos significativos nestes inteiros (3) é menor que o número de dígitos significativos nas coordenadas (4), o Spatial Extender não poderá converter estes inteiros novamente para as coordenadas originais ou derivar deles valores que sejam consistentes com o sistema de coordenadas ao qual estas coordenadas pertencem. Porém, se o fator de escala for 100, os números inteiros resultantes que o DB2 Spatial Extender armazena serão 1001 e 2003 — valores que podem ser convertidos de novo para as coordenadas originais ou dos quais as coordenadas compatíveis podem ser derivadas.

Unidades para Valores de Deslocamento e Fatores de Escala

Caso você utilize um sistema de referência espacial existente ou crie um novo, as unidades dos valores de deslocamento e os fatores de escala variarão dependendo do tipo de sistema de coordenadas que estiver utilizando. Por exemplo, se estiver utilizando um sistema de coordenadas geográficas, os valores estão em unidades angulares, tais como, graus decimais; se você estiver utilizando um sistema de coordenadas projetadas, os valores estarão em unidades lineares, como metros ou pés.

Tarefas Relacionadas:

- "Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema" na página 68

Criando um Sistema de Referência Espacial

Se nenhum dos sistemas de referência espacial fornecidos com o DB2 Spatial Extender funcionar com seus dados, será necessário criar um novo sistema de referência espacial.

Procedimento:

Para criar um novo sistema de referência espacial:

1. Escolha a interface.

   Você pode criar um sistema de referência espacial de uma das formas a seguir:

- Utilize a janela Criar Sistema de Referência Espacial no Centro de Controle do DB2. Consulte a ajuda on-line para obter informações adicionais sobre como utilizar esta janela.
- Emita o comando **db2se create_srs** no processador da linha de comandos do db2se. Para obter informações adicionais, consulte "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127.
- Execute um aplicativo que chame o procedimento armazenado db2se.ST_create_srs. Para obter informações adicionais, consulte "ST_create_srs" na página 240.

2. Especifique um SRID (Spatial Reference System ID) apropriado:
   - Para dados geodésicos em uma representação redonda da Terra, especifique um valor de SRID no intervalo de 200000318 a 2000001000.
   - Para dados espaciais em uma representação plana da Terra, especifique um SRID que ainda não esteja definido.
3. Decida o grau de precisão que deseja. Você pode:
   - Especifique as extensões da área geográfica com que está trabalhando e os fatores de escala que deseja utilizar com os dados de coordenadas. O Spatial Extender utiliza as extensões especificadas e calcula o deslocamento.

     Você pode especificar extensões de uma das seguintes formas:
     - Escolha **Extensões** na janela Criar Sistema de Referência Espacial do Centro de Controle.
     - Forneça os parâmetros apropriados para o comando **db2se create_srs** ou o procedimento armazenado db2se.ST_create_srs.
   - Especifique os valores de deslocamento (obrigatório para o Spatial Extender converter valores negativos em valores positivos) e os fatores de escala (obrigatório para o Spatial Extender converter valores decimais em inteiros). Utilize este método quando precisar seguir critérios estritos para exatidão ou precisão.

     Você pode especificar valores de deslocamento e fatores de escala de uma das seguintes formas:
     - Escolha **Deslocamento** na janela Criar Sistema de Referência Espacial do Centro de Controle
     - Forneça os parâmetros apropriados para o comando **db2se create_srs** ou o procedimento armazenado db2se.ST_create_srs.

     Para obter informações adicionais, consulte "Fatores de Conversão que Transformam Dados de Coordenadas em Inteiros" na página 73.
4. Calcule as informações de conversão das quais o Spatial Extender precisa para converter dados de coordenadas em inteiros positivos e forneça estas informações por meio da interface escolhida. Estas informações diferem de acordo com o método escolhido na etapa 3.
   - Se você escolheu o método "Extensões" na etapa 3, precisará calcular as seguintes informações:
     - Fatores de escala. Se qualquer uma das coordenadas com a qual você está trabalhando for valor decimal, calcule fatores de escala (consulte "Calculando Fatores de Escala" na página 77. Os fatores de escala são números que, quando multiplicados por coordenadas e medidas decimais, resultam em inteiros com pelo menos o mesmo número de dígitos significativos das coordenadas e medidas originais. Se as coordenadas forem números inteiros, os fatores de escala poderão ser definidos em 1. Se as coordenadas forem valores decimais, o fator de escala deverá ser

definido em um número que converta a parte decimal para um valor inteiro. Por exemplo, se as unidades de coordenadas forem metros e a precisão dos dados for de 1 cm, você precisará de um fator de escala de 100.

– Valores mínimo e máximo das coordenadas e medidas. (Consulte "Determinando Coordenadas e Medidas Mínimas e Máximas" na página 78 para obter informações adicionais).

• Se você escolheu o método "Deslocamento" na etapa 3, precisará calcular as seguintes informações:

– Valores de deslocamento

Se os dados de coordenada incluem números ou medidas negativas, será necessário especificar os valores de deslocamento que você deseja utilizar. Um deslocamento é um número que é subtraído de todas as coordenadas, deixando somente valores positivos como restante. Se estiver trabalhando com coordenadas positivas, defina todos os valores de deslocamento como 0. Se não estiver trabalhando com coordenadas positivas, selecione um deslocamento que, quando aplicado nos dados de coordenadas, resulte em inteiros menores que o maior valor inteiro positivo (9,007,199,254,740,992). (Consulte "Calculando Valores de Deslocamento" na página 79 para obter informações adicionais).

– Fatores de escala

Se alguma coordenada das localizações que estiver representando for número decimal, determine quais fatores de escala serão utilizados e insira esses fatores de escala na janela Criar Sistema de Referência Espacial. Consulte "Calculando Fatores de Escala" na página 77.

5. Envie o comando **db2se create_srs** ou o procedimento armazenado db2se.ST_create_srs.

Por exemplo, o comando a seguir cria um sistema de referência espacial denominado mysrs:

```
db2se create_srs mydb -srsName \"mysrs\"
 -srsID 100 -xScale 10 -coordsysName
\"GCS_North_American_1983\"
```

Para obter informações adicionais sobre como executar um aplicativo que chama o procedimento armazenado db2se.ST_create_srs, consulte "ST_create_srs" na página 240.

Depois de criar o sistema de referência espacial, associe-o a uma coluna espacial com uma das seguintes tarefas:

- "Criando Colunas Espaciais" na página 83
- "Registrando Colunas Espaciais" na página 85

Conceitos Relacionados:

- "Fatores de Conversão que Transformam Dados de Coordenadas em Inteiros" na página 73
- "Sistemas de Coordenadas" na página 59
- "Sistemas de Referência Espacial" na página 67

Tarefas Relacionadas:

- "Calculando Fatores de Escala" na página 77
- "Determinando Coordenadas e Medidas Mínimas e Máximas" na página 78
- "Calculando Valores de Deslocamento" na página 79
- "Criando Colunas Espaciais" na página 83

- "Registrando Colunas Espaciais" na página 85
- "Creating a spatial reference system: Spatial Extender help"
- "Registering a spatial column with a spatialreference system: Spatial Extender help"

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127
- "ST_create_srs" na página 240

Calculando Fatores de Escala

Se você criar um sistema de referência espacial e qualquer uma das coordenadas com as quais você está trabalhando for valor decimal, calcule os fatores de escala apropriados para suas coordenadas e medidas. Os fatores de escala são números que, quando multiplicados por coordenadas e medidas decimais, resultam em inteiros com pelo menos o mesmo número de dígitos significativos das coordenadas e medidas originais.

Pré-requisitos:

Antes de calcular os fatores de escala que funcionarão com seus dados, assegure-se de que tenha entendido as diretrizes para escolher "Fatores de Conversão que Transformam Dados de Coordenadas em Inteiros" na página 73.

Procedimento:

Para calcular os fatores de escala:

1. Determine quais coordenadas X e Y são, ou provavelmente serão, números decimais. Suponha, por exemplo, que das várias coordenadas X e Y com as quais você estará lidando, você determina que três delas são números decimais: 1.23, 5.1235 e 6.789.
2. Encontre a coordenada decimal que tenha a precisão decimal mais longa. Em seguida, determine por qual potência de 10 esta coordenada pode ser multiplicada para gerar um inteiro de igual precisão. Por exemplo, das três coordenadas decimais no exemplo atual, 5,1235 tem a precisão decimal mais longa. Multiplicá-lo por 10 à quarta potência (10000) gerará o inteiro 51235.
3. Determine se o inteiro produzido pela multiplicação recém-descrita é menor que 2^{53}. 51235 não é muito grande. Mas, suponha que, além de 1.23, 5.11235 e 6.789, o intervalo de coordenadas X e Y inclua um quarto valor decimal, 10000000006.789876. Como esta precisão decimal da coordenada é maior do que as outras três, você pode multiplicar *esta* coordenada—não 5.1235—por uma potência de 10. Para convertê-la em um inteiro, você poderia multiplicá-la por 10 elevado a seis (1000000). Mas o valor resultante, 10000000006789876, é maior do que 2^{53}. Se o DB2 Spatial Extender tentasse armazená-lo, os resultados seriam imprevisíveis.

   Para evitar este problema, selecione uma potência de 10 que, quando multiplicada pela coordenada original, gere um número decimal que o DB2 Spatial Extender possa truncar para um inteiro armazenável, com perda mínima de precisão. Neste caso, você pode selecionar 10 à quinta potência (100000). Multiplicando 100000 por 10000000006.789876 resulta em 1000000000678987.6. O DB2 Spatial Extender arredonda este número para 1000000000678988, reduzindo ligeiramente sua precisão.

Depois de calcular fatores de escala, é necessário determinar os valores de extensão (consulte "Determinando Coordenadas e Medidas Mínimas e Máximas". Em seguida, envie o comando **db2se create_srs** ou o procedimento armazenado db2se.ST_create_srs.

Conceitos Relacionados:

- "Fatores de Conversão que Transformam Dados de Coordenadas em Inteiros" na página 73
- "Sistemas de Referência Espacial" na página 67

Tarefas Relacionadas:

- "Determinando Coordenadas e Medidas Mínimas e Máximas" na página 78

Determinando Coordenadas e Medidas Mínimas e Máximas

Determine as coordenadas e medidas mínimas e máximas se você decidir especificar transformações de extensões quando criar um sistema de referência espacial.

Pré-requisitos:

Utilize este procedimento para determinar as coordenadas e medidas mínimas e máximas se você:

- Decidir criar um novo sistema de referência espacial porque nenhum dos sistemas de referência espacial fornecidos com o DB2 Spatial Extender funcionam com seus dados. Para obter informações adicionais, consulte "Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema" na página 68.
- Decidir utilizar transformações de extensões para converter suas coordenadas.

Procedimento:

Para determinar as coordenadas e medidas mínimas e máximas das localizações que deseja representar:

- Determine as coordenadas X mínimas e máximas.

  Para encontrar a coordenada X mínima, identifique-a no seu domínio que esteja na extremidade oeste. (Se a localização ficar à oeste do ponto de origem, essa coordenada será um valor negativo.) Para encontrar a coordenada X máxima, identifique-a no seu domínio que esteja na extremidade leste. Por exemplo, se estiver representando poços de petróleo e cada um estiver definido por um par de coordenadas X e Y, a coordenada X que indica a localização do poço que está na extremidade oeste será a coordenada X mínima e a que indica a localização na extremidade leste será a coordenada X mínima.
- Determine as coordenadas Y mínimas e máximas.

  Para encontrar a coordenada Y mínima, identifique-a no domínio que esteja na extremidade sul. (Se a localização ficar ao sul do ponto de origem, essa coordenada será um valor negativo.) Para determinar a coordenada Y máxima, localize-a no domínio que esteja na extremidade norte.
- Determine as coordenadas Z mínimas e máximas.

  A coordenada Z mínima é a maior das coordenadas de profundidade e a coordenada Z a maior das coordenadas de altura.
- Determine as medidas mínimas e máximas

Para incluir medidas nos dados espaciais, determine qual medida tem o maior valor numérico e qual tem o menor.

Para tipos de recursos múltiplos, como polígonos múltiplos, verifique se escolheu o ponto mais extremo no polígono mais extremo na direção que está calculando. Por exemplo, se estiver tentando identificar a coordenada X mínima, identifique a coordenada na extremidade X oeste do polígono que está na extremidade oeste no polígono múltiplo.

Depois de determinar os valores de extensões, se qualquer uma das coordenadas for valor decimal, será necessário calcular fatores de escala (consulte "Calculando Fatores de Escala" na página 77. Caso contrário, envie o comando **db2se create_srs** ou o procedimento armazenado db2se.ST_create_srs.

Tarefas Relacionadas:

- "Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema" na página 68
- "Calculando Fatores de Escala" na página 77

Calculando Valores de Deslocamento

Se você criar um sistema de referência espacial e os dados de suas coordenadas incluírem números ou medidas negativas, será necessário especificar os valores de deslocamento que você deseja utilizar. Um deslocamento é um número que é subtraído de todas as coordenadas, deixando somente valores positivos como restante. Você pode aprimorar o desempenho de operações espaciais quando as coordenadas forem inteiros positivos em vez de números ou medidas negativas..

Pré-requisitos:

Você especifica valores de deslocamento se os dados de suas coordenadas incluírem números ou medidas negativas.

Procedimento:

Para calcular valores de deslocamento para as coordenadas com que está trabalhando:

1. Determine as menores coordenadas X, Y e Z negativas dentro do intervalo de coordenadas das localizações que deseja representar. Se os seus dados tiverem que incluir medidas negativas, determine a menor destas medidas. Consulte "Determinando Coordenadas e Medidas Mínimas e Máximas" na página 78.
2. Opcional, mas recomendado: Indique ao DB2 Spatial Extender que o domínio que abrange as localizações de seu interesse é maior do que realmente é. Assim, depois de gravar os dados sobre essas localizações em uma coluna espacial, você poderá incluir dados sobre localizações de recursos novos, à medida que forem incluídos em distâncias distantes do domínio, sem a necessidade de substituir o sistema de referência espacial por outro.

   Para cada coordenada e medida identificada na etapa 1, adicione uma quantidade igual a cinco para dez por cento da coordenada ou medida. O resultado é referido como um *valor aumentado*. Por exemplo, se a menor coordenada X negativa for –100, você pode acrescentar –5 a ela, gerando um valor aumentado de –105.Posteriormente, quando você criar o sistema de referência espacial, indique que a menor coordenada X é –105, em vez do valor verdadeiro de –100.O DB2 Spatial Extender, então, interpretará –105 como o limite mais a oeste de seu domínio.

3. Encontre um valor que, quando subtraído do valor X aumentado, reste zero, este é o valor de deslocamento para as coordenadas X. O DB2 Spatial Extender subtrai esse número de todas as coordenadas X para produzir somente valores positivos.

   Por exemplo, se o valor X aumentado for –105, será necessário subtrair –105 dele para obter 0. O DB2 Spatial Extender irá, então, subtrair –105 de todas as coordenadas X que estão associadas aos recursos que estão sendo representados.Como nenhuma destas coordenadas é maior que –100, todos os valores que resultam da subtração serão positivos.
4. Repita a etapa 3 para o valor Y aumentado, o valor Z aumentado e a medida aumentada.

Depois de calcular valores de deslocamento, crie um sistema de referência espacial (consulte “Criando um Sistema de Referência Espacial” na página 74.

Tarefas Relacionadas:

- “Determinando Coordenadas e Medidas Mínimas e Máximas” na página 78
- “Criando um Sistema de Referência Espacial” na página 74

Capítulo 9. Configurando Colunas Espaciais

Na preparação da obtenção de dados espaciais para um projeto, você não apenas escolhe ou cria um sistema de coordenadas e sistema de referência espacial, como também produz uma ou mais colunas de tabela para conter os dados. Este capítulo:

- Indica que os resultados das consultas das colunas podem ser processados graficamente, e fornece diretrizes para escolher os tipos de dados para as colunas
- Descreve a tarefa para fornecer as colunas
- Descreve a tarefa para tornar as colunas acessíveis para ferramentas que podem exibir seu conteúdo na forma gráfica

Colunas Espaciais

Colunas Espaciais com Conteúdo Exibível

Quando você utiliza uma ferramenta de visualização como ArcExplorer para DB2®, para consultar uma coluna espacial, a ferramenta retorna resultados na forma de exibição gráfica, por exemplo, um mapa de limites de áreas ou o layout de um sistema rodoviário. Algumas ferramentas de visualização requerem que todas as linhas da coluna utilizem o mesmo sistema de referência espacial. A maneira que você reforça esta limitação é registrar a coluna com um sistema de referência espacial.

Tipos de Dados Espaciais

Quando você ativa um banco de dados para operações espaciais, o DB2 Spatial Extender fornece ao banco de dados uma hierarquia de tipos de dados estruturados. A Figura 12 na página 82 apresenta esta hierarquia. Nesta figura, os tipos que podem ser instanciados tem o plano de fundo em branco; os tipos que não podem ser instanciados possuem um plano de fundo sombreado.

Os tipos de dados instanciáveis são ST_Point, ST_LineString, ST_Polygon, ST_GeomCollection, ST_MultiPoint, ST_MultiPolygon e ST_MultiLineString.

Os tipos de dados que não são instanciáveis são ST_Geometry, ST_Curve, ST_Surface, ST_MultiSurface e ST_MultiCurve.

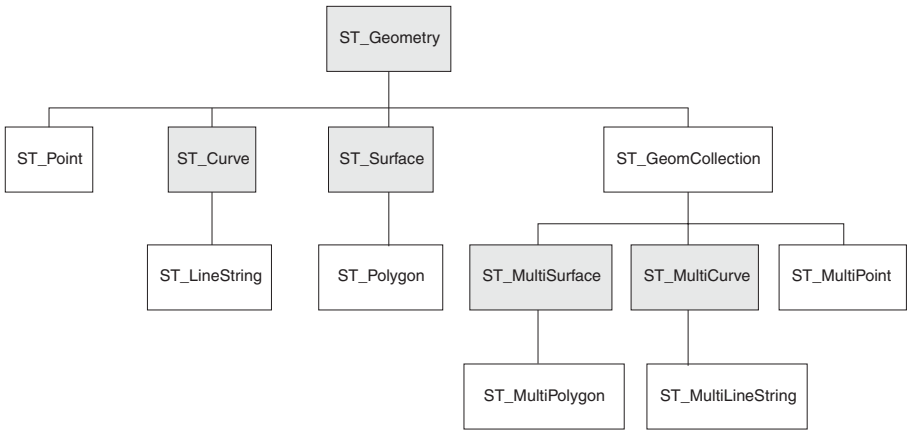


Figura 12. Hierarquia dos Tipos de Dados Espaciais. Tipos de Dados em Caixas Brancas Podem Ser Instanciados. Os tipos de dados nomeados nas caixas sombreadas não são instanciáveis.

A hierarquia na Figura 12 inclui:

- Tipos de dados para recursos geográficos que podem ser captados formando uma única unidade; por exemplo, residências individuais e lagos isolados.
- Tipos de dados para recursos geográficos que são compostos de várias unidades ou componentes; por exemplo, sistemas de canais e grupos de ilhas em um lago.
- Um tipo de dados para recursos geográficos de todos os tipos.

Tipos de Dados para Recursos de uma Única Unidade

Utilize ST_Point, ST_LineString e ST_Polygon para armazenar coordenadas que definem o espaço ocupado por recursos que podem ser percebidos na formação de uma única unidade:

- Utilize ST_Point quando desejar indicar o ponto no espaço que é ocupado por um recurso geográfico separado. O recurso pode ser muito pequeno, como um poço de água; pode ser muito grande, como uma cidade; ou pode ser intermediário, como um conjunto de prédios ou um parque. Em cada caso, o ponto no espaço pode estar localizado na interseção de uma linha de coordenada leste-oeste (por exemplo, uma paralela) e uma norte-sul (por exemplo, um meridiano). Um item de dados ST_Point inclui uma coordenada X e uma coordenada Y que definem essa interseção. A coordenada X indica onde a interseção está situada na linha leste-oeste; a coordenada Y indica onde a interseção está situada na linha norte-sul.
- Utilize ST_Linestring para coordenadas que definem o espaço que é ocupado por recursos lineares; por exemplo, ruas, canais e canalizações.
- Utilize ST_Polygon quando desejar indicar a extensão do espaço coberto por um recurso multifacetado; por exemplo, uma região, uma floresta ou um habitat natural. Um item de dados de ST_Polygon consiste de coordenadas que definem o limite de tal recurso.

Em alguns casos, ST_Polygon e ST_Point podem ser utilizados para o mesmo recurso. Por exemplo, suponha que você precise de informações espaciais sobre um complexo de apartamentos. Se desejar representar o ponto no espaço onde cada prédio no complexo está localizado, utilize ST_Point para armazenar as coordenadas X e Y que definem cada um desses pontos. De outra forma, se desejar representar a área ocupada pelo complexo como um todo, utilize ST_Polygon para armazenar as coordenadas que definem o limite desta área.

Tipos de Dados para Recursos de Multi-unidades

Utilize ST_MultiPoint, ST_MultiLineString e ST_MultiPolygon para armazenar coordenadas que definem espaços ocupados por recursos compostos de várias unidades:

- Utilize ST_MultiPoint quando estiver representando recursos compostos de unidades cujas localizações sejam referidas individualmente por uma coordenada X e uma coordenada Y. Por exemplo, considere uma tabela cujas linhas representam uma cadeia de ilhas. Foi identificada a coordenada X e a coordenada Y para cada ilha. Se deseja que a tabela inclua estas coordenadas e as coordenadas de cada cadeia como um todo, defina uma coluna ST_MultiPoint para conter estas coordenadas.
- Utilize ST_MultiLineString quando estiver representando recursos compostos de unidades lineares e desejar armazenar as coordenadas para as localizações destas unidades e a localização de cada recurso como um todo. Por exemplo, considere uma tabela cujas linhas representam um sistema de rios. Se deseja que a tabela inclua coordenadas para as localizações do sistema e de seus componentes, defina uma coluna ST_MultiLineString para conter estas coordenadas.
- Utilize ST_MultiPolygon quando estiver representando recursos compostos de unidades multifacetadas e desejar armazenar as coordenadas para as localizações destas unidades e a localização de cada recurso como um todo. Por exemplo, considere uma tabela cujas linhas representam áreas rurais e as fazendas em cada área. Se deseja que a tabela inclua coordenadas para as localizações das áreas e fazendas, defina uma coluna ST_MultiPolygon para conter estas coordenadas.

Multiunidade não significa uma coleção de entidades individuais. Multiunidade se refere a uma agregação das partes que compõem o todo.

Um Tipo de Dados para Todos os Recursos

Você pode utilizar ST_Geometry quando não tiver certeza sobre qual dos outros tipos de dados utilizar. Como ST_Geometry é a raiz da hierarquia à qual os outros tipos de dados pertencem, uma coluna ST_Geometry pode conter o mesmo tipo de itens de dados que as colunas dos outros tipos de dados podem conter.

Atenção:

Se estiver planejando utilizar o geocoder fornecido, DB2SE_USA_GEOCODER, para gerar dados para uma coluna espacial, a coluna deverá ser do tipo ST_Point ou ST_Geometry. No entanto, algumas ferramentas de visualização não suportam as colunas ST_Geometry, mas apenas as colunas às quais um subtipo apropriado de ST_Geometry foi atribuído.

Tarefas Relacionadas:

- “Registrando Colunas Espaciais” na página 85
- “Criando Colunas Espaciais” na página 83

Criando Colunas Espaciais

Esta tarefa faz parte de uma tarefa maior: ″Configurando Recursos Espaciais para um Projeto.″ Depois de escolher um sistema de coordenadas e determinar qual sistema de referência espacial utilizar para seus dados, crie uma coluna espacial em uma tabela existente ou importe dados espaciais para uma nova tabela.

Pré-requisitos:

Configurando Colunas Espaciais

Antes de criar uma coluna espacial, seu ID do usuário deve conter as autorizações necessárias para a instrução DB2 SQL CREATE TABLE ou ALTER TABLE. O ID do usuário deve ter pelo menos uma das seguintes autoridades ou privilégios:

- Autoridade SYSADM ou DBADM no banco de dados no qual reside a tabela que contém a coluna
- Autoridade CREATETAB no banco de dados e o privilégio USE no espaço de tabelas, além de um dos seguintes:
  - Autoridade IMPLICIT_SCHEMA no banco de dados, se o esquema implícito ou explícito do índice não existir
  - Privilégio CREATEIN no esquema, se o nome do esquema do índice se referir a um esquema existente
- Privilégio ALTER na tabela a ser alterada
- Privilégio CONTROL na tabela a ser alterada
- Privilégio ALTERIN sobre o esquema da tabela

Procedimento:

Você pode fornecer a seu banco de dados colunas espaciais de uma dentre várias maneiras:

- Utilize a instrução CREATE TABLE do DB2 para criar uma tabela e para incluir uma coluna espacial nessa tabela.
- Utilize a instrução ALTER TABLE do DB2 para incluir uma coluna espacial em uma tabela existente.
- Utilize a janela Criar Coluna Espacial no Centro de Controle do DB2. Abra a janela Colunas Espaciais a partir de uma tabela. Consulte a ajuda on-line para obter informações adicionais sobre como utilizar esta janela.
- Se estiver importando dados espaciais de um arquivo de formas, utilize o DB2 Spatial Extender para criar uma tabela e para fornecer a esta tabela uma coluna para conter os dados. Consulte “Importando Dados de Formas para uma Tabela Nova ou Existente” na página 88.
- Se estiver importando dados espaciais de um arquivo de transferência SDE, utilize o DB2 Spatial Extender para criar uma tabela, para fornecer a esta tabela uma coluna para conter os dados e para tornar a coluna acessível a ferramentas de visualização. Consulte “Importando Dados de Transferência SDE para uma Tabela Nova ou Existente” na página 90.

Próxima tarefa: “Registrando Colunas Espaciais” na página 85

Tarefas Relacionadas:

- “Importando Dados de Formas para uma Tabela Nova ou Existente” na página 88
- “Importando Dados de Transferência SDE para uma Tabela Nova ou Existente” na página 90
- “Registrando Colunas Espaciais” na página 85
- “Gerando Aplicativos para o DB2 Spatial Extender” na página 135
- “Creating a spatial column: Spatial Extender help”

Referência Relacionada:

- “ALTER TABLE statement” na publicação *SQL Reference, Volume 2*
- “CREATE TABLE statement” na publicação *SQL Reference, Volume 2*

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127

Registrando Colunas Espaciais

Talvez você queira registrar uma coluna espacial nas seguintes situações:

- Acesso por ferramentas de visualização

  Se desejar que determinadas ferramentas de visualização — por exemplo, ArcExplorer para DB2 — gerem exibições gráficas dos dados em uma coluna espacial, deve garantir a integridade dos dados da coluna. Pode-se fazer isso impondo uma limitação que requer que todas as linhas da coluna utilizem o mesmo sistema de referência espacial. Para impor esta limitação, registre a coluna, especificando seu nome e o sistema de referência espacial que se aplica a ela.
- Acesso por índices espaciais

  Utilize o mesmo sistema de coordenadas para todos os dados em uma coluna espacial na qual você deseja criar um índice para assegurar que o índice espacial retorne os resultados corretos. Você registra uma coluna espacial para limitar todos os dados para utilizarem o mesmo sistema de referência espacial e, de forma correspondente, o mesmo sistema de coordenadas.

Pré-requisitos:

Antes de registrar uma coluna espacial, seu ID de usuário deve conter uma das seguintes formas de autorização:

- Autoridade SYSADM ou DBADM no banco de dados no qual reside a tabela que contém a coluna.
- O privilégio CONTROL ou ALTER nesta tabela.

Se estiver utilizando o pocessador da linha de comandos do db2se ou um programa aplicativo para importar dados de um arquivo de transferência SDE, você poderá fazer com que o DB2 Spatial Extender crie e registre automaticamente uma coluna para conter os dados. Neste caso, seu ID de usuário deve conter a autoridade SYSADM ou DBADM no banco de dados.

Procedimento:

Você pode registrar uma coluna espacial de uma das seguintes formas:

- Utilize as janelas Colunas Espaciais e Selecionar Sistema de Referência Espacial do Centro de Controle do DB2 para registrar a coluna.
- Emita o comando **db2se register_spatial_column**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.ST_register_spatial_column.
- Se desejar importar dados espaciais de um arquivo de transferência SDE, você poderá utilizar a janela Importar Dados Espaciais do Centro de Controle, o comando **import_sde** ou o procedimento armazenado db2gse.ST_import_sde para criar uma tabela com uma coluna espacial, para registrar esta coluna e para importar os dados para a coluna.

Consulte a coluna SRS_NAME na exibição DB2GSE.GSE_GEOMETRY_COLUMNS para verificar o sistema de referência espacial escolhido para uma determinada coluna, depois de registrá-la.

Tarefas Relacionadas:

- "Gerando Aplicativos para o DB2 Spatial Extender" na página 135

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127
- "ST_register_spatial_column" na página 272

Capítulo 10. Ocupando Colunas Espaciais

Depois de criar colunas espaciais e registrar as que serão acessadas por estas ferramentas de visualização, você está pronto para ocupar as colunas com dados espaciais. Há três maneiras de fornecer os dados: importá-los; utilizar um geocoder para derivá-los dos dados de negócios; ou utilizar funções espaciais para criá-los ou para derivá-los dos dados de negócios ou outros dados espaciais. Este capítulo:

- Apresenta o conceito e a tarefa para importar dados espaciais para o banco de dados, e o conceito e a tarefa para exportar dados espaciais para arquivos que os aplicativos podem utilizar.
- Descreve o geocoding e apresenta as tarefas para configurar operações de geocoding, configurar geocoders para serem executados automaticamente e executar geocoders no modo batch.

Como Importar e Exportar Dados Espaciais

Esta seção descreve o conceito de importação e exportação de dados, e apresenta as seguintes tarefas:

- Importar dados espaciais para uma nova tabela ou para uma tabela ou exibição existente
- Exportar dados espaciais para arquivos que os aplicativos podem utilizar

Sobre Importação e Exportação de Dados Espaciais

Você pode utilizar o DB2® Spatial Extender para trocar dados espaciais entre o banco de dados e as origens de dados externas. Mais precisamente, você pode importar dados espaciais de origens externas transferindo-os para seu banco de dados em arquivos, chamados *arquivos de troca de dados*. Você também pode exportar dados espaciais de seu banco de dados para arquivos de troca de dados a partir dos quais as origens externas podem adquiri-los. Esta seção apresenta algumas das razões para importar e exportar dados espaciais e descreve a natureza dos arquivos de troca de dados suportados pelo DB2 Spatial Extender.

Razões para importar e exportar dados espaciais:

Ao importar dados espaciais, você pode obter uma grande quantidade de informações espaciais que já estão disponíveis na indústria. Exportando-os você pode disponibilizá-los em um formato de arquivo padrão para aplicativos existentes. Considere estes cenários:

- Seu banco de dados contém dados espaciais que representam seus escritórios de vendas, clientes e outros assuntos comerciais. Você deseja suplementar estes dados com dados espaciais que representam seu ambiente cultural da organização — cidades, ruas, pontos de interesse, e assim por diante. Os dados que você deseja estão disponíveis a partir de um mapa do fornecedor. Você pode utilizar o DB2 Spatial Extender para importá-los de um arquivo de troca de dados oferecido pelo fornecedor.
- Você deseja migrar dados espaciais de um sistema Oracle para o ambiente do DB2. Você continuou utilizando um utilitário Oracle para gravar os dados em um arquivo de troca de dados. Utilize, então, o DB2 Spatial Extender para importar os dados deste arquivo para o banco de dados que foi ativado para operações espaciais.

- Você não está conectado ao DB2, e deseja utilizar um geobrowser para mostrar aos clientes apresentações visuais de informações espaciais. O navegador precisa apenas de arquivos para trabalhar; ele não precisa estar conectado a um banco de dados. É possível utilizar o DB2 Spatial Extender para exportar os dados para um arquivo de troca de dados e, então, utilizar um navegador para processar os dados em formato visual.

Arquivos Shape e Arquivos de Transferência SDE:

O DB2 Spatial Extender suporta dois tipos de arquivos de troca de dados: arquivos de formas e arquivos de transferência SDE. O termo *arquivo shape* significa um conjunto de arquivos com o mesmo nome, mas com extensões de arquivo diferentes. A coleção pode incluir até quatro arquivos. Eles são:

- Um arquivo que contém dados espaciais em *formato de formas*, um formato padrão da indústria desenvolvido pelo ESRI. Esses dados geralmente são chamados de *formato de dados*. A extensão de um arquivo que contém formato de dados é .shp.
- Um arquivo que contém dados de negócios que pertence a localizações definidas por formato de dados. A extensão deste arquivo é .dbf.
- Um arquivo que contém um índice para formato de dados. A extensão deste arquivo é .shx.
- Um arquivo que contém uma especificação do sistema de coordenadas no qual os dados em um arquivo .shp estão baseados. A extensão deste arquivo é .prj.

Os arquivos shape geralmente são utilizados para importar dados originados em sistemas de arquivos e para exportar dados para arquivos dentro de sistemas de arquivos.

Ao utilizar o DB2 Spatial Extender para importar dados de formato, você recebe pelo menos um arquivo .shp. Na maioria dos casos, você também pode receber um ou mais dos outros três tipos de arquivos shape.

Arquivos de transferência SDE geralmente são utilizados para importar dados originados em bancos de dados ESRI. Cada arquivo inclui dados espaciais, um sistema de referência espacial para esses dados e dados de negócios. Os dados espaciais, cujo formato é propriedade do ESRI, destinam-se a uma coluna de tabela que foi registrada no catálogo do DB2 Spatial Extender. Os dados de negócios são direcionados para outras colunas na tabela às quais a coluna registrada pertence.

Importando Dados Espaciais

Esta seção fornece uma visão geral das tarefas para importar formato de dados e dados de transferência SDE para o banco de dados. A seção inclui referências cruzadas para informações específicas (por exemplo, processos e parâmetros) para executar essas tarefas.

Importando Dados de Formas para uma Tabela Nova ou Existente

Você pode importar formato de dados para uma tabela ou exibição existente ou pode criar uma tabela e importar formato de dados para ela em uma única operação. Mais especificamente, você pode:

- Importar os dados shape para uma coluna espacial em uma tabela existente, uma exibição updatetable existente ou uma exibição existente na qual um acionador INSTEAD OF para INSERTs está definido

- Criar automaticamente uma tabela com uma coluna espacial e importar os formato de dados para esta coluna

Pré-requisitos:

Antes de importar formato de dados para uma tabela ou exibição existente, seu ID de usuário deve conter uma das seguintes formas de autorização:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela ou exibição
- Privilégio CONTROL na tabela ou exibição
- Privilégio INSERT na tabela ou exibição
- O privilégio SELECT na tabela ou exibição (necessário somente se a tabela incluir uma coluna de ID que não seja uma coluna IDENTITY)
- Privilégios para acessar os diretórios aos quais os arquivos de entrada e arquivos de erros pertencem
- Privilégios de leitura nos arquivos de entrada e privilégios de gravação em arquivos de erros

Antes de começar a criar uma tabela automaticamente e importar formato de dados para ela, seu ID de usuário deve conter uma das seguintes formas de autorização:

- Autoridade SYSADM, DBADM ou CREATETAB no banco de dados que contém a tabela
- Uma das seguintes permissões:
  - Privilégio CREATEIN no esquema ao qual a tabela pertence (necessário somente quando o esquema já existir)
  - Autoridade IMPLICIT_SCHEMA no banco de dados que contém a tabela (necessário quando o esquema especificado para a tabela não existe atualmente)
- Privilégios para acessar os diretórios aos quais os arquivos de entrada e arquivos de erros pertencem
- Privilégios de leitura nos arquivos de entrada e privilégios de gravação em arquivos de erros

Procedimento:

Você pode importar formato de dados de uma das seguintes maneiras:

- Utilize a janela Importar Dados de Formas do Centro de Controle do DB2.
- Emita o comando **db2se import_shape**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.ST_import_shape.

Recomendação:

Você pode melhorar o desempenho do processo de importação explorando recursos disponíveis no DB2. Por exemplo, ao importar dados para uma tabela existente ou para uma tabela que você criar, defina a tabela como NOT LOGGED INITIALLY, especificando os parâmetros apropriados de criação de tabela.

Conceitos Relacionados:

- "Sobre Importação e Exportação de Dados Espaciais" na página 87

Tarefas Relacionadas:

- "Importando Dados de Transferência SDE para uma Tabela Nova ou Existente" na página 90
- "Gerando Aplicativos para o DB2 Spatial Extender" na página 135

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127
- "ST_import_shape" na página 259

Importando Dados de Transferência SDE para uma Tabela Nova ou Existente

Você pode importar dados de transferência SDE para uma tabela existente ou pode criar uma tabela e importar dados de transferência SDE para ela em uma única operação. Mais especificamente, você pode:

- Importar dados de transferência SDE para uma tabela existente que inclua uma coluna espacial que já esteja registrada no catálogo do DB2 Spatial Extender. Os dados de transferência podem incluir dados espaciais para a coluna e dados de negócios para outras colunas na tabela.
- Criar automaticamente uma tabela que tenha uma coluna espacial, registrar esta coluna no catálogo e importar dados de transferência SDE para esta coluna e também para outras colunas da tabela.

Pré-requisitos:

Antes de importar dados para uma coluna em uma tabela ou exibição existente, seu ID de usuário deve conter uma das seguintes formas de autorização:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela ou exibição
- Privilégio CONTROL na tabela ou exibição
- Privilégios INSERT e SELECT na tabela ou exibição

Antes de iniciar a operação de criar uma tabela automaticamente e importar formato de dados para ela, seu ID de usuário deve conter uma das seguintes formas de autorização:

- Autoridade SYSADM, DBADM ou CREATETAB no banco de dados que contém a tabela
- Uma das seguintes permissões:
  - Privilégio CREATEIN no esquema ao qual a tabela pertence (necessário somente quando o esquema já existir)
  - Autoridade IMPLICIT_SCHEMA no banco de dados que contém a tabela (necessário quando o esquema especificado para a tabela atualmente não existe)

Procedimento:

Você pode importar dados de transferência SDE de uma das seguintes formas:

- Utilize a janela Importar do Centro de Controle do DB2.
- Emita o comando **db2se import_sde**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.GSE_import_sde.

Para obter informações sobre como executar estas ações, consulte as fontes que estão listadas em "Tarefas relacionadas" no final desta discussão.

Conceitos Relacionados:

- "Sobre Importação e Exportação de Dados Espaciais" na página 87

Tarefas Relacionadas:

- "Importando Dados de Formas para uma Tabela Nova ou Existente" na página 88
- "Gerando Aplicativos para o DB2 Spatial Extender" na página 135

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127
- "GSE_import_sde" na página 230

Exportando Dados Espaciais

Esta seção fornece uma visão geral das tarefas para exportar dados espaciais para arquivos de formas e de transferência SDE. A seção inclui referências cruzadas para informações específicas (por exemplo, processos e parâmetros) para executar essas tarefas.

Exportando Dados para um Arquivo Modelo

Você pode exportar dados espaciais retornados em resultados de consultas para um arquivo modelo. Os dados podem vir de várias origens, como uma tabela base, uma junção ou união de várias tabelas, conjuntos de resultados retornados quando você consulta suas exibições ou a saída de uma função espacial.

Se existir um arquivo para o qual você deseja exportar dados, o DB2 Spatial Extender poderá anexar os dados a este arquivo. Se este arquivo não existir, você poderá utilizar o DB2 Spatial Extender para criar um.

Pré-requisitos:

Antes de exportar dados para um arquivo modelo, seu ID de usuário deve conter os seguintes privilégios:

- O privilégio para executar uma subseleção que retorna os resultados que você deseja exportar
- O privilégio para gravar no diretório em que reside o arquivo para o qual você exportará dados
- O privilégio para criar um arquivo para conter os dados exportados (obrigatório se o arquivo ainda não existir)

Para saber quais são estes privilégios e como obtê-los, consulte o administrador do banco de dados.

Procedimento:

Você pode exportar dados para um arquivo modelo de qualquer uma das seguintes maneiras:

- Inicie a exportação a partir da janela Exportar Arquivo de Formas do Centro de Controle do DB2.

- Emita o comando **db2se export_shape** a partir do processador de linha de comandos do db2se.
- Execute um aplicativo que chame o procedimento armazenado db2gse.ST_export_shape.

Para obter informações sobre como executar estas ações, consulte as fontes que estão listadas em "Tarefas relacionadas" no final desta discussão.

Exportando Dados para um Arquivo de Transferência SDE

Você pode exportar uma tabela que contém dados espaciais para um arquivo de transferência SDE. A tabela não pode conter mais de uma coluna espacial. Além disso, esta coluna deve ser registrada no catálogo do DB2 Spatial Extender. Se a tabela contiver dados de negócios, estes dados serão exportados junto com os dados espaciais. Você pode exportar todas as linhas na tabela ou um subconjunto de linhas. Para exportar um subconjunto, especifique uma cláusula WHERE que identifica o subconjunto.

Pré-requisitos:

Antes de exportar dados para um arquivo de transferência SDE, seu ID do usuário deve conter as seguintes permissões:

- Autoridade SYSADM OU DBADM.
- O privilégio SELECT na tabela que será exportada.
- O privilégio para gravar no diretório em que reside o arquivo para o qual você exportará dados

Restrições:

- Você pode exportar somente uma coluna espacial em cada operação de exportação.
- As colunas que forem exportadas devem ter tipos de dados suportados pelo formato SDE.
- A tabela deve conter exatamente uma coluna espacial.
- Esta coluna deve ser registrada no catálogo do DB2 Spatial Extender.
- Você não pode anexar a arquivos SDE existentes.

Procedimento:

Você pode exportar dados espaciais e de negócios para um arquivo de transferência SDE de qualquer uma das seguintes formas:

- Utilize a janela Exportar Arquivos SDE do Centro de Controle do DB2.
- Emita o comando **db2se export_sde**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.GSE_export_sde.

Conceitos Relacionados:

- "Sobre Importação e Exportação de Dados Espaciais" na página 87

Tarefas Relacionadas:

- "Gerando Aplicativos para o DB2 Spatial Extender" na página 135

Referência Relacionada:

- "Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos" na página 127
- "GSE_export_sde" na página 228

Como Utilizar um Geocoder

Esta seção descreve o conceito de geocoding e apresenta as seguintes tarefas:

- Definir o trabalho a ser executado por um geocoder; por exemplo, especificar quantos registros o geocoder deve processar antes de uma consolidação ser emitida
- Configurar um geocoder para geocodificar os dados assim que os dados são incluídos ou atualizados em uma tabela.
- Executar um geocoder no modo batch

Geocoders e Geocoding

Os termos *geocoder* e *geocoding* são utilizados em vários contextos. Esta discussão classifica estes contextos, para que os significados dos termos possam ser compreendidos sempre que você consultá-los. A discussão define *geocoder* e *geocoding*, descreve os modos nos quais um geocoder opera, descreve uma maior atividade à qual o geocoding pertence e resume as tarefas do usuário relacionadas ao geocoding.

No DB2® Spatial Extender, um geocoder é uma função escalar que converte dados existentes (a entrada da função) em dados que você pode compreender em termos espaciais (a saída da função). Geralmente, os dados existentes são dados relacionais que descrevem ou nomeiam uma localização. Por exemplo, o geocoder que é fornecido com o DB2 Spatial Extender, DB2SE_USA_GEOCODER, converte endereços dos Estados Unidos em dados de ST_Point. O DB2 Spatial Extender também pode suportar geocoders fornecidos pelo usuário e pelo fornecedor; e sua entrada e saída não precisam ser semelhantes às do DB2SE_USA_GEOCODER. Para ilustrar: Um geocoder do fornecedor pode converter endereços em coordenadas que o DB2 não armazena, mas grava em um arquivo. O outro pode converter o número de um escritório em um edifício comercial em coordenadas que definem a localização do escritório no edifício ou pode converter o identificador de uma prateleira em um warehouse em coordenadas que definem a localização da prateleira no warehouse.

Em outros casos, os dados existentes convertidos por um geocoder podem ser dados espaciais. Por exemplo, um geocoder fornecido pelo usuário pode converter coordenadas X e Y em dados que estão de acordo com um dos tipos de dados do DB2 Spatial Extender.

No DB2 Spatial Extender, *geocoding* é apenas a operação na qual um geocoder converte sua entrada em saída — converter endereços em coordenadas, por exemplo.

Modos:

Um geocoder opera em dois modos:

- No *modo batch*, um geocoder tenta, em uma única operação, converter todas as suas entradas a partir de uma única tabela. Por exemplo, no modo batch, o

DB2SE_USA_GEOCODER tenta converter todos os endereços em uma única tabela (ou, como alternativa, todos os endereços em um subconjunto de linhas especificado na tabela).

- No *modo automático*, um geocoder converte dados assim que são inseridos ou atualizados em uma tabela. O geocoder é ativado pelos disparos INSERT e UPDATE que estão definidos na tabela.

Processos de Geocoding:

Geocoding é uma das diversas operações pelas quais o conteúdo de uma coluna espacial em uma tabela do DB2 é derivado de outros dados. Esta discussão se refere a estas operações coletivamente como um *processo de geocoding*. Os processos de geocoding podem variar de geocoder para geocoder. Por exemplo, o DB2SE_USA_GEOCODER pesquisa arquivos de endereços conhecidos para determinar se cada endereço que ele recebe como entrada corresponde com um endereço conhecido em um determinado grau. Como os endereços conhecidos são semelhantes ao material de referência que as pessoas procuram quando fazem pesquisa, estes endereços são coletivamente chamados de *dados de referência.* Outros geocoders podem não precisar de dados de referência; eles podem verificar sua entrada de outras formas. O processo de geocoding do qual o DB2SE_USA_GEOCODER participa é o seguinte:

1. O DB2SE_USA_GEOCODER executa operações para as quais ele foi designado:
   a. O DB2SE_USA_GEOCODER analisa cada endereço que recebe como entrada.
   b. O DB2SE_USA_GEOCODER pesquisa nos dados de referência os nomes de ruas que, em um determinado grau, assemelham-se ao nome da rua no endereço analisado. Ele limita sua pesquisa à ruas na área designada pelo CEP do endereço.
   c. Se a pesquisa for bem-sucedida, o DB2SE_USA_GEOCODER determina se algum endereço nas ruas que ele encontrou corresponde com o endereço analisado em um determinado grau.
   d. Se o DB2SE_USA_GEOCODER encontrar uma correspondência, ele efetua geocode do endereço analisado. Caso contrário, retorna nulo.
2. Se o DB2SE_USA_GEOCODER efetuar geocode do endereço analisado, o DB2 colocará as coordenadas resultantes em uma coluna espacial designada.
3. Se o DB2SE_USA_GEOCODER estiver efetuando geocode no modo batch, o DB2 Spatial Extender emitirá uma consolidação (a) sempre que o DB2SE_USA_GEOCODER concluir o processamento de um determinado número de registros de entrada ou (b) depois que o DB2SE_USA_GEOCODER concluir o processamento de todas as suas entradas.

As Tarefas do Usuário:

No DB2 Spatial Extender, as tarefas relacionadas a geocoding são:

- Prescrever como determinadas partes do processo de geocoding devem ser executadas para uma coluna espacial especificada; por exemplo, definir o grau mínimo ao qual os nomes de ruas em registros de entrada e nomes de ruas em dados de referência devem corresponder; definir o grau mínimo ao qual os endereços em registros de entrada e os endereços em dados de referência devem corresponder; e determinar quantos registros devem ser processados antes de cada consolidação. Esta tarefa pode ser referida como *configurando o geocoding* ou *configurando operações de geocoding*.

- Especificar que deve ser efetuado o geocode automático dos dados sempre que eles forem incluídos ou atualizados em uma tabela. Quando ocorre o geocoding automático, as instruções que o usuário especificou ao configurar as operações de geocoding serão efetivadas (exceto as instruções que envolvem consolidações; elas se aplicam somente ao geocoding em batch). Esta tarefa é referida como *configurando em geocoder para execução automática.*
- Executar um geocoder no modo batch. Se o usuário já configurou operações de geocoding, suas instruções permanecerão em efeito durante cada sessão em batch, a menos que o usuário substitua-as. Se o usuário não configurou operações de geocoding antes de uma determinada sessão, ele poderá especificar que elas devem ser efetivadas, configurando-as para essa sessão específica. Esta tarefa pode ser referida como *executar um geocoder no modo batch* e *executar geocoding no modo batch.*

Configurando Operações de Geocoding

O DB2 Spatial Extender permite definir, antecipadamente, o trabalho que precisa ser feito quando um geocoder é chamado. Por exemplo, você pode especificar:

- Para qual coluna o geocoder deve fornecer dados.
- Se a entrada que o geocoder lê a partir de uma tabela ou exibição deve ser limitada a um subconjunto de linhas na tabela ou exibição.
- O intervalo ou o número de registros nos quais o geocoder deve efetuar geocode em sessões em batch em uma unidade de trabalho
- Requisitos para operações específicas de geocoder. Por exemplo, o DB2SE_USA_GEOCODER pode efetuar geocode somente nos registros que correspondam às suas contrapartes nos dados de referência em um grau especificado ou maior. Este grau é chamado de *score mínimo de correspondência.*

Você *deve* especificar os parâmetros descritos anteriormente antes de configurar o geocoder para ser executado no modo automático. Desse ponto em diante, sempre que o geocoder for chamado (não apenas automaticamente, mas também para execuções em batch), as operações de geocoding serão executadas de acordo com suas especificações. Por exemplo, se você especificar que deve ser efetuado geocode em 45 registros no modo batch em cada unidade de trabalho, será emitida uma consolidação depois de ser efetuado o geocode a cada quarenta e cinco registros. (Exceção: você pode substituir suas especificações para sessões individuais de geocoding em batch.)

Você *não* precisa estabelecer padrões para operações de geocoding antes de executar o geocoder no modo batch. Em vez disso, no momento em que iniciar uma sessão em batch, você pode especificar como as operações devem ser realizadas durante toda a execução. Se você estabelecer padrões para sessões em batch, poderá substituí-las, conforme necessário, por sessões individuais.

Pré-requisitos:

Antes de definir as operações de geocoding para um determinado geocoder, seu ID de usuário deve conter uma das seguintes formas de autorização:

- Autoridade SYSADM ou DBADM no banco de dados que contém as tabelas nas quais o geocoder irá operar
- O privilégio SELECT e o privilégio CONTROL ou UPDATE em cada tabela na qual o geocoder opera

Procedimento:

Você pode configurar as operações de geocoding de uma das seguintes maneiras:

- Chame-o a partir da janela Configurar Geocoding do Centro de Controle do DB2.
- Emita o comando **db2se setup_gc**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.ST_setup_geocoding.

Para obter informações sobre como executar estas ações, consulte as fontes que estão listadas em "Tarefas relacionadas" no final desta discussão.

Recomendações:

- Quando o DB2SE_USA_GEOCODER lê um registro de dados de endereços, ele tenta corresponder esse registro a uma contraparte nos dados de referência. Explicando mais detalhadamente: a forma que ele procede é a seguinte: Primeiro, ele pesquisa nos dados de referência as ruas cujo CEP seja igual ao CEP do registro. Se ele encontra um nome de rua que seja semelhante ao do registro em um grau mínimo especificado, ou em um grau maior do que este mínimo, ele continua procurando um endereço inteiro. Se ele encontra um endereço inteiro que seja semelhante ao do registro em um grau mínimo especificado, ou em um grau maior do que este mínimo, ele efetua geocode no registro. Se ele não encontra o endereço, retorna nulo.

  O grau mínimo ao qual os nomes de ruas devem corresponder é referido como *sensibilidade de ortografia*. O grau mínimo ao qual todos os endereços devem corresponder é chamado de *score mínimo de correspondência*. Por exemplo, se a sensibilidade de ortografia for de 80, a correspondência entre os nomes de ruas deve conter, pelo menos, 80 por cento de precisão antes que o geocoder pesquise o endereço inteiro. Se o score mínimo de correspondência for 60, a correspondência entre os endereços deve ser, pelo menos, 60 por cento de precisão antes que o geocoder efetue geocode do registro.

  Você pode especificar qual deve ser a sensiblidade de ortografia e o score mínimo de correspondência. Lembre-se de que você precisa ajustá-los. Por exemplo, suponha que a sensibilidade de ortografia e o score mínimo de correspondência sejam 95. Se os endereços nos quais você deseja efetuar geocode não foram validados com atenção, as correspondências com 95 por cento de precisão provavelmente não serão encontradas. Como resultado, o geocoder provavelmente retornará nulo quando processar estes registros. Neste caso, é recomendável reduzir a sensibilidade de ortografia e o score mínimo de correspondência e executar o geocoder novamente. Os scores recomendados para sensibilidade de ortografia e o score mínimo de correspondência são 70 e 60, respectivamente.
- Conforme mencionado no início desta discussão, você pode determinar se a entrada que o geocoder lê, a partir de uma tabela ou exibição, deve ser limitada a um subconjunto de linhas na tabela ou exibição. Por exemplo, considere os seguintes cenários:
  - Você chama o geocoder para efetuar geocode nos endereços em uma tabela no modo batch. Infelizmente, o score mínimo de correspondência é muito alto, fazendo o geocoder retornar nulo quando ele processa a maioria dos endereços. Você reduz o score mínimo de correspondência quando executar o geocoder novamente. Para limitar sua entrada aos endereços nos quais não foi efetuado geocode, especifique se ele deve selecionar somente as linhas que contenham o nulo retornado anteriormente.
  - O geocoder seleciona somente as linhas que foram incluídas depois de uma determinada data.

  - O geocoder seleciona somente as linhas que contêm endereços em uma determinada área; por exemplo, um bloco de regiões ou um estado.
- Conforme mencionado no início desta discussão, você pode determinar o número de registros que o geocoder deve processar em sessões em batch em uma unidade de trabalho. Você pode fazer o geocoder processar o mesmo número de registros em cada unidade de trabalho ou pode fazer ele processar todos os registros de uma tabela em uma única unidade de trabalho. Se você escolher a última alternativa, lembre-se de que:
  - Você tem menos controle sobre o tamanho da unidade de trabalho do que os recursos alternativos anteriores. Conseqüentemente, você não pode controlar quantos bloqueios estão retidos ou quantas entradas de log são criadas durante a operação do geocoder.
  - Se o geocoder encontrar um erro que precise de uma reversão, será necessário executar o geocoder para executar todos os registros novamente. O custo resultante com recursos pode ser caro se a tabela for extremamente grande e o erro e a reversão ocorrerem após a maioria dos registros ter sido processada.

Configurando um Geocoder para Execução Automática

Você pode configurar um geocoder para converter dados automaticamente assim que os dados são incluídos ou atualizados em uma tabela.

Pré-requisitos:

Antes de configurar um geocoder para execução automática:

- Você deve configurar as operações de geocoding para cada coluna espacial que deve ser ocupada pela saída do geocoder.
- Seu ID de usuário deve conter as seguintes formas de autorização:
  - Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual os disparos que chamam o geocoder serão definidos
  - Um ou mais privilégios nesta tabela:
    - O privilégio CONTROL.
    - Se você não tiver o privilégio CONTROL, precisará dos privilégios ALTER, SELECT e UPDATE.
  - Os privilégios necessários para criar disparos nesta tabela.

Procedimento:

Existem três formas de configurar o geocoding automático:

- Faça isso a partir da janela Configurar Geocoding ou da janela Geocoding do Centro de Controle do DB2.
- Emita o comando **db2se enable_autogc**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.ST_enable_autogeocoding.

Para obter informações sobre como executar estas ações, consulte as fontes que estão listadas em “Tarefas relacionadas” no final desta discussão.

Recomendações:

- Você pode configurar um geocoder para execução automática antes de chamá-lo no modo batch. Portanto, é possível que o geocoding automático preceda o geocoding em batch. Se isso ocorrer, o geocoding em batch provavelmente

envolverá o processamento dos mesmos dados que foram processados automaticamente. Esta redundância não resultará em duplicação de dados porque, quando os dados espaciais são gerados duas vezes, a segunda geração de dados substitui a primeira. De qualquer modo, ele pode degradar o desempenho.

- Antes de decidir se efetuará o geocode nos dados de endereços em uma tabela no modo batch ou no modo automático, considere que:
  - O desempenho é melhor no geocoding em batch do que no geocoding automático. Uma sessão em batch é aberta com uma inicialização e termina com uma limpeza. No geocoding automático, é efetuado geocode em cada item de dados em uma única operação que começa com inicialização e é concluída com uma limpeza.
  - Em geral, uma coluna espacial ocupada por meio de geocoding automático provavelmente seja mais atualizada do que uma coluna espacial ocupada por meio de geocoding em batch. Depois de uma sessão em batch, os dados de endereço podem ficar acumulados e permanecer sem geocode até a próxima sessão. Mas, se o geocoding automático já estiver ativado, será efetuado o geocode nos dados de endereço assim que forem armazenados no banco de dados.

Executando um Geocoder no Modo Batch

Você pode chamar um geocoder para ser executado no modo batch; ou seja, tentar, em uma única operação, converter vários registros em dados espaciais que são colocados em uma coluna específica.

A qualquer momento, antes de executar um geocoder para ocupar uma determinada coluna espacial, você pode configurar as operações de geocoding para essa coluna. Configurar as operações envolve a especificação de alguns requisitos que devem ser atendidos quando o geocoder for executado. Por exemplo, suponha que você exija que o DB2 Spatial Extender emita uma consolidação a cada 100 registros de entrada que forem processados pelo geocoder. Ao configurar as operações, especifique 100 como o número requerido.

Quando estiver pronto para executar o geocoder, você poderá substituir qualquer um dos valores especificados durante a configuração das operações. Suas substituições permanecerão em efeito somente durante a execução.

Se você não configurar as operações, sempre que estiver pronto para executar o geocoder, deverá especificar como os requisitos devem ser atendidos durante a execução.

Pré-requisitos:

Antes de executar um geocoder no modo batch, seu ID de usuário deve conter uma das seguintes formas de autorização:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela em cujos dados devem ser efetuados geocode
- O privilégio CONTROL ou UPDATE nesta tabela
-

Você também precisa do privilégio SELECT nesta tabela para que possa especificar o número de registros a serem processados antes de cada consolidação. Se você especificar cláusulas WHERE para limitar as linhas nas quais o geocoder deve

operar, também poderá requerer o privilégio SELECT nas tabelas e exibições que forem referidas nestas cláusulas. Consulte o administrador do banco de dados.

Restrições:

Procedimento:

Você pode chamar um geocoder para execução no modo batch em qualquer uma das seguintes formas:

- Chame-o a partir da janela Executar Geocoding do Centro de Controle do DB2.
- Emita o comando **db2se run_gc**.
- Execute um aplicativo que chame o procedimento armazenado db2gse.ST_run_geocoding.

Capítulo 11. Utilizando Índices e Exibições para Acessar Dados Espaciais

Antes de consultar colunas espaciais, você pode criar índices e exibições que facilitarão o acesso a elas. Este capítulo:

- Descreve a natureza dos índices que o Spatial Extender utiliza para expedir o acesso aos dados espaciais
- Explica como criar esses índices
- Explica como utilizar exibições para acessar dados espaciais

Tipos de Índices Espaciais

O bom desempenho da consulta está relacionado ao fato de ter índices eficientes definidos nas colunas das tabelas base em um banco de dados. O desempenho da consulta está diretamente relacionado à rapidez com que os valores na coluna podem ser encontrados durante a consulta. As consultas que utilizam um índice podem ser executadas mais rapidamente e podem fornecer uma melhora significativa no desempenho.

As consultas espaciais geralmente são consultas que envolvem duas ou mais dimensões. Por exemplo, em uma consulta espacial, talvez você queira saber se um ponto está incluído em uma área (polígono). Devido à natureza multidimensional de consultas espaciais, a indexação de árvore B nativa do DB2® é ineficiente para estas consultas.

As consultas espaciais podem utilizar os seguintes tipos de índices:

- Índices de grade espaciais

  A tecnologia de indexação do DB2 Spatial Extender utiliza a *indexação de grades*, designada para indexar dados espaciais multidimensionais a colunas espaciais de índice. O DB2 Spatial Extender fornece um índice de grade que é otimizado para dados bidimensionais em uma projeção plana da Terra.
- Índices geodésicos de Voronoi

  O DB2 Geodetic Extender fornece suporte para um novo método de acesso espacial que permite criar índices em colunas que contêm dados geodésicos multidimensionais. Um índice geodésico de Voronoi é mais adequado do que um índice de grade para dados geodésicos porque trata a Terra como uma esfera contínua sem distorções em torno dos pólos ou bordas no meridiano de 180 graus.

Conceitos Relacionados:

- "Índices Geodésicos de Voronoi" na página 177
- "Índices de Grade Espaciais" na página 102

Tarefas Relacionadas:

- "Criando Índices Geodésicos de Voronoi" na página 181
- "Criando Índices de Grades Espaciais" na página 108

Referência Relacionada:

- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Índices de Grade Espaciais

Os índices melhoram o desempenho da consulta de aplicativos, principalmente quando a tabela ou tabelas consultadas contêm muitas linhas. Se você criar índices apropriados para o otimizador de consulta escolher para executar sua consulta, poderá reduzir significativamente o número de linhas a serem processadas.

O DB2 Spatial Extender fornece um índice de grade que é otimizado para dados bidimensionais. O índice é criado nas dimensões X e Y de uma geometria.

Os seguintes aspectos de um índice de grade são úteis para entender:

- A geração do índice
- A utilização de funções espaciais em uma consulta
- Como uma consulta utiliza um índice de grade espacial

Geração de Índices de Grade Espaciais

O Spatial Extender gera um índice de grade espacial utilizando o MBR (Minimum Bounding Rectangle) de uma geometria. Para a maioria das geometrias, o MBR é um retângulo que engloba a geometria. Para obter detalhes adicionais sobre MBRs, consulte “ST_MBR” na página 433.

Um índice de grade espacial divide uma região em grades quadradas lógicas com um tamanho fixo que você especifica quando cria o índice. O índice espacial é construído em um coluna espacial por meio da criação de uma ou mais entradas para as interseções de cada MBR da geometria com as células da grade. Uma entrada de índice consiste no identificador da célula da grade, no MBR da geometria e no identificador interno da linha que contém a geometria.

Você pode definir até três níveis de índice espacial (níveis de grade). A utilização de vários níveis de grade é útil porque permite otimizar o índice para diferentes tamanhos de dados espaciais. Para obter informações adicionais, consulte “Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade” na página 104.

Se uma geometria cruzar quatro ou mais células da grade, a geometria será promovida para o próximo nível mais alto. Em geral, as geometrias maiores serão indexadas nos níveis mais altos. Se uma geometria cruzar 10 ou mais células da grade no maior tamanho de grade, será utilizado um nível de índice de estouro definido pelo sistema. Este nível de estouro evita a geração de excessivas entradas de índices. Para obter melhor desempenho, defina seus tamanhos de grades para evitar a utilização deste nível de estouro.

Por exemplo, se existirem vários níveis de grades, o algoritmo de indexação tentará utilizar o menor nível de grade possível para fornecer a melhor resolução para os dados indexados. Quando uma geometria cruza mais de quatro células de grade em determinado nível, ela é promovida para o próximo nível maior (desde que exista outro nível). Portanto, um índice espacial que tem os três níveis de grade de 10.0, 100.0 e 1000.0 primeiro cruzará cada geometria com a grade de nível 10.0. Se uma geometria cruzar com mais de quatro células da grade de tamanho 10.0, ela será promovida e cruzada com a grade de nível 100.0. Se mais de quatro interseções resultarem no nível 100.0, a geometria será promovida para o nível 1000.0. Se mais de 10 interseções resultarem no nível 1000.0, a geometria será indexada no nível do estouro.

Utilização de Funções Espaciais em uma Consulta

O otimizador do DB2 UDB considera a utilização de um índice de grade espacial quando uma consulta contém as seguintes funções em sua cláusula WHERE:

- ST_Contains
- ST_Crosses
- ST_Distance
- ST_EnvIntersects
- EnvelopesIntersect
- ST_Equals
- ST_Intersects
- ST_MBRIntersects
- ST_Overlaps
- ST_Touches
- ST_Within

Para obter informações adicionais, consulte "Funções que Utilizam Índices para Otimizar Consultas" na página 124.

Como uma Consulta Utiliza um Índice de Grade Espacial

Quando o otimizador de consulta escolhe um índice de grade espacial , a execução da consulta utiliza o seguinte processo de filtragem de várias etapas:

1. Determine as células da grade que cruzam a janela de consulta. A *janela de consulta* é a geometria de seu interesse e que você especifica como o segundo parâmetro em uma função espacial (consulte os exemplos abaixo).
2. Varra o índice em busca de entradas que possuem identificadores de células de grade correspondentes.
3. Compare os valores MBR de geometria nas entradas de índice com a janela de consulta e descarte os valores que estão fora da janela de consulta.
4. Execute análise adicional, se necessário. O conjunto de geometrias candidato das etapas anteriores pode passar por análise adicional para determinar se elas atendem a função espacial (ST_Contains, ST_Distance, etc). A função espacial EnvelopesIntersect omite esta etapa e, geralmente, possui o melhor desempenho.

Os exemplos de consultas espaciais a seguir possuem um índice de grade espacial na coluna C.GEOMETRY:

```
SELECT name
FROM counties AS c
WHERE EnvelopesIntersect(c.geometry, -73.0, 42.0, -72.0, 43.0, 1) = 1

SELECT name
FROM counties AS c
WHERE ST_Intersects(c.geometry, :geometry2) = 1
```

No primeiro exemplo, os quatro valores de coordenadas definem a janela de consulta. Estes valores de coordenadas especificam os cantos inferior esquerdo e superior direito (42.0 –73.0 e 43.0 –72.0) de um retângulo.

No segundo exemplo, o Spatial Extender calcula o MBR da geometria especificada pela variável do host :geometry2 e utiliza-o como a janela de consulta.

Quando criar um índice de grade espacial, você deve especificar tamanhos de grades apropriados para os tamanhos de janelas de consulta mais comuns que seu

aplicativo espacial provavelmente utilizará. Se um tamanho de grade for maior, as entradas de índice para geometrias que estão fora da janela de consulta deverão ser varridas porque residem nas células de grade que cruzam a janela de consulta e essas varreduras extras reduzem o desempenho. No entanto, um tamanho de grade menor pode gerar mais entradas de índice para cada geometria e entradas de índice adicionais devem ser varridas, o que também reduz o desempenho da consulta.

O DB2 Spatial Extender fornece um utilitário Orientador de Índice que analisa os dados da coluna espacial e sugere tamanhos de grades apropriados para tamanhos de janela de consulta típica. Para obter informações adicionais, consulte "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113.

Conceitos Relacionados:

- "Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade" na página 104
- "Tipos de Índices Espaciais" na página 101
- "Ajustando Índices de Grade Espaciais com o Orientador de Índice—Visão Geral" na página 112

Tarefas Relacionadas:

- "Criando Índices de Grades Espaciais" na página 108

Referência Relacionada:

- "Instrução CREATE INDEX para um Índice de Grade Espacial" na página 110

Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade

Utilize o Orientador de Índice para determinar tamanhos de grade apropriados para os índices de grade espaciais, pois esta é a melhor forma de ajustar os índices e tornar as consultas espaciais mais eficientes.Esta seção fornece conceitos para ajudá-lo a entender os efeitos de diferentes níveis e tamanhos de grades.

Número de Níveis de Grade

Você pode ter até três níveis de grade. No entanto, para cada nível em um índice de grade espacial, é executada uma pesquisa de índice separada durante uma consulta espacial. Portanto, se houver mais níveis de grade, a consulta será menos eficiente.

Se os valores na coluna espacial tiverem quase o mesmo tamanho relativo, utilize um único nível de grade. No entanto, uma coluna espacial típica não contém geometrias do mesmo tamanho relativo, mas as geometrias em uma coluna espacial podem ser agrupadas de acordo com o tamanho. Você deve corresponder seus níveis de grade com estes agrupamentos de geometrias.

Por exemplo, suponha que você tenha uma tabela de terrenos de uma região com uma coluna espacial que contenha agrupamentos de pequenos terrenos urbanos cercados por grandes terrenos rurais. Como os tamanhos dos terrenos podem ser colocados em dois grupos (pequenos terrenos urbanos e grandes terrenos rurais), você pode especificar dois níveis de grade para o índice de grade espacial.

Tamanhos de Células de Grade

A regra geral é reduzir os tamanhos de grades o máximo possível para obter a melhor resolução ao reduzir o número de entradas de índice.

- Um valor pequeno deve ser utilizado para o menor tamanho da grade para otimizar todo o índice de geometrias pequenas na coluna. Isto evita a sobrecarga de avaliar geometrias que não estão na área de pesquisa. No entanto, o menor tamanho da grade também gera o maior número de entradas de índices. Conseqüentemente, o número de entradas de índices processadas no momento da consulta aumenta, conforme a quantidade de armazenamento necessário para o índice. Estes fatores reduzem o desempenho geral.
- Ao utilizar tamanhos maiores de grades, o índice pode ser otimizado para geometrias maiores. Os maiores tamanhos de grades geram menos entradas de índices para grandes geometrias do que o menor tamanho de grade. Conseqüentemente, os requisitos de armazenamento para o índice são reduzidos, aumentando o desempenho geral.

A figura a seguir mostra os efeitos de diferentes tamanhos de grades.

A Figura 13 mostra um mapa de lotes de terra, cada lote representado por uma geometria de polígono. O retângulo preto representa uma janela de consulta. Suponha que você deseja localizar todas as geometrias cujo MBR cruze a janela de consulta. A Figura 13 mostra que 28 geometrias (realçadas em rosa) possuem um MBR que cruza a janela de consulta.

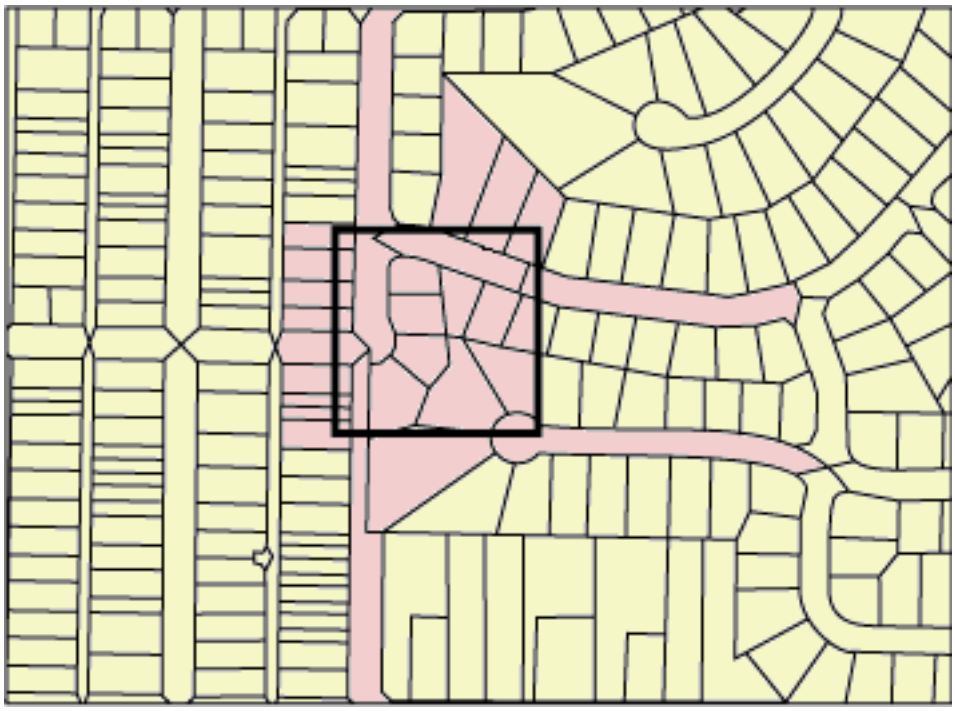


Figura 13. Lotes de Terra em uma Vizinhança

A Figura 14 na página 106 mostra um tamanho de grade pequeno (25) que fornece um ajuste aproximado da janela de consulta.

- A consulta retorna apenas as 28 geometrias que estão realçadas, mas a consulta deve examinar e descartar três geometrias adicionais cujos MBRs cruzam a janela de consulta.

- Este tamanho de grade pequeno resulta em muitas entradas de índice por geometria. Durante a execução, a consulta acessa todas as entradas de índice para estas 31 geometrias. A Figura 14 mostra 256 células de grade que sobrepõem a janela de consulta. No entanto, a execução da consulta acessa 578 entradas de índice porque muitas geometrias são indexadas com as mesmas células de grade.

Para esta janela de consulta, este tamanho de grade pequeno resulta em um número excessivo de entradas de índice a serem varridas.

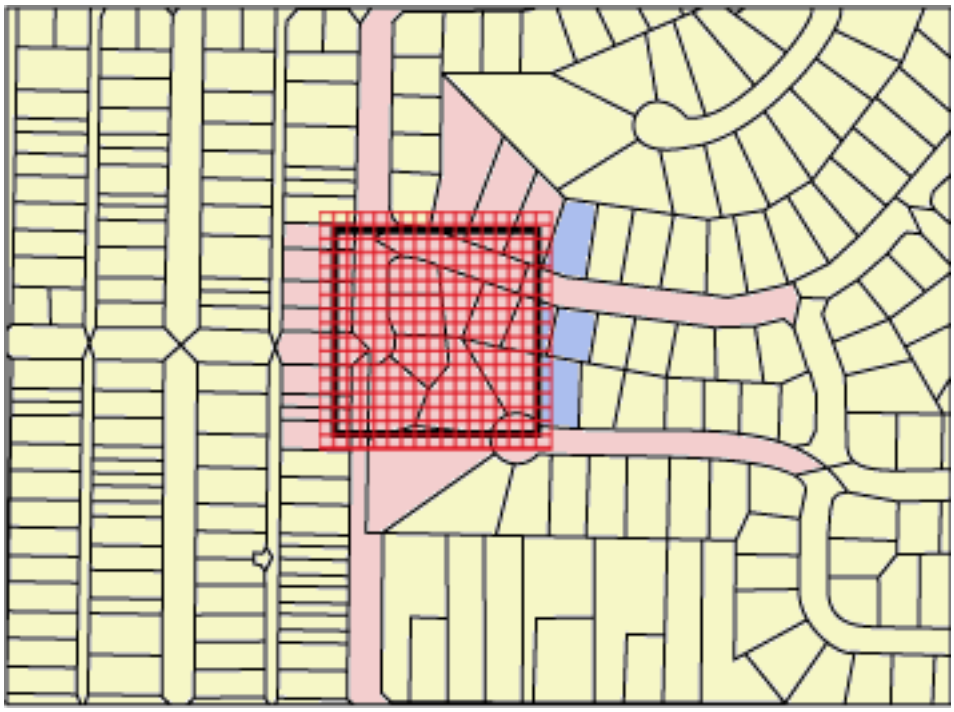


Figura 14. Tamanho de Grade Pequeno (25) em Lotes de Terra

A Figura 15 na página 107 mostra um tamanho de grade grande (400) que inclui uma área consideravelmente maior com muito mais geometrias do que a janela de consulta.

- Este tamanho de grade grande resulta em apenas uma entrada de índice por geometria, mas a consulta deve examinar e descartar 59 geometrias adicionais cujos MBRs cruzam a célula de grade.
- Durante a execução, a consulta acessa todas as entradas de índice para as 28 geometrias que cruzam a janela de consulta, além das entradas de índice para as 59 geometrias adicionais, para um total de 112 entradas de índice.

Para esta janela de consulta, este tamanho de grade grande resulta em um número excessivo de geometrias a serem examinadas.

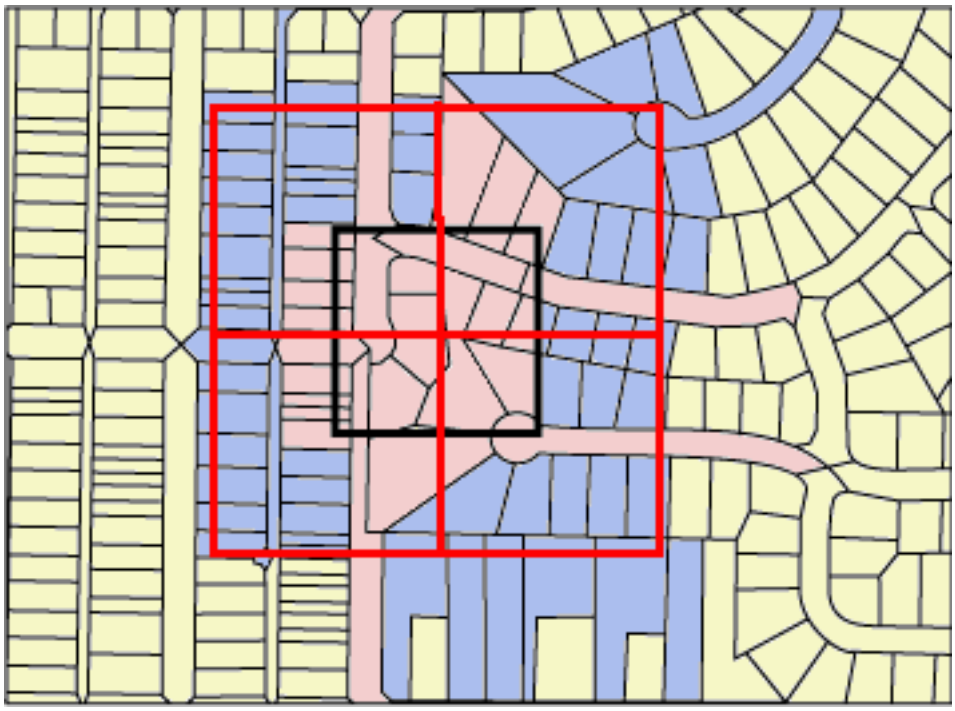


Figura 15. Tamanho de Grade Grande (400) em Lotes de Terra

A Figura 16 na página 108 mostra um tamanho de grade médio (100) que fornece um ajuste aproximado da janela de consulta.

- A consulta retorna apenas as 28 geometrias que estão realçadas, mas a consulta deve examinar e descartar cinco geometrias adicionais cujos MBRs cruzam a janela de consulta.
- Durante a execução, a consulta acessa todas as entradas de índice para as 28 geometrias que cruzam a janela de consulta, além das entradas de índice para as 5 geometrias adicionais, para um total de 91 entradas de índice.

Para esta janela de consulta, esta tamanho médio de grade é o melhor porque ele resulta em muito menos entradas de índice do que o tamanho de grade pequeno e a consulta examina menos geometrias adicionais do que o tamanho de grade grande.

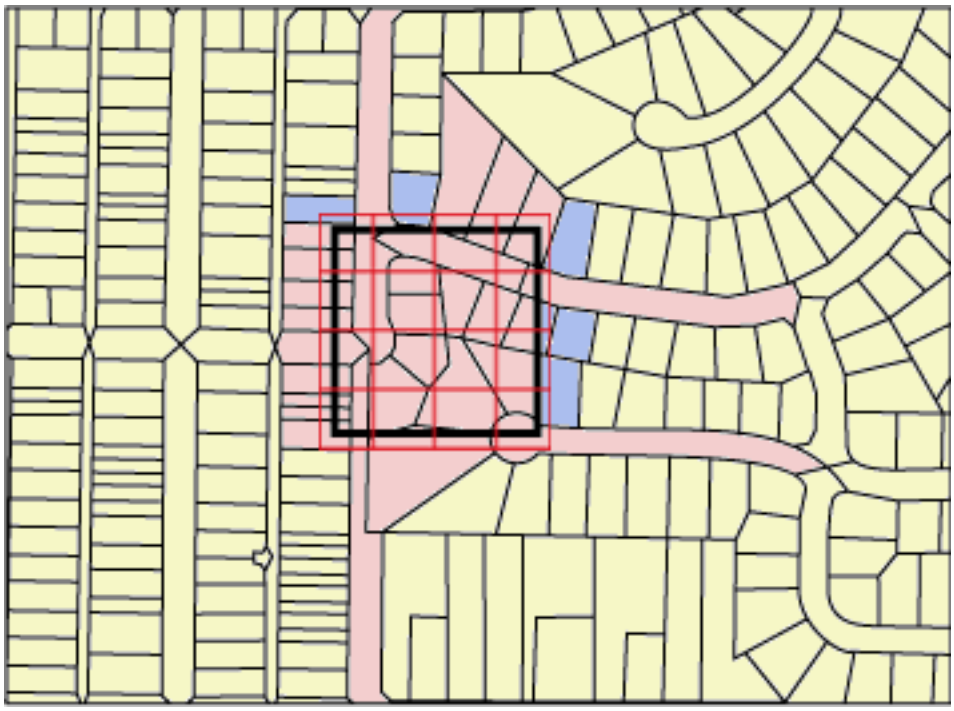


Figura 16. Tamanho de Grade Médio (100) em Lotes de Terra

Conceitos Relacionados:

- "Índices de Grade Espaciais" na página 102
- "Ajustando Índices de Grade Espaciais com o Orientador de Índice—Visão Geral" na página 112

Tarefas Relacionadas:

- "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113
- "Analisando Estatísticas de Índice de Grade Espacial" na página 114
- "Criando Índices de Grades Espaciais" na página 108

Referência Relacionada:

- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Criando Índices de Grades Espaciais

Você cria índices de grade espaciais para aprimorar o desempenho de consultas em colunas espaciais.

Quando criar um índice de grade espacial, forneça as seguintes informações:

- Um nome
- O nome da coluna espacial na qual ele será definido
- Os tamanhos de grades (consulte "Índices de Grade Espaciais" na página 102). A combinação dos três tamanhos de grades ajuda a otimizar o desempenho reduzindo o número total de entradas de índice e o número de entradas de índice que precisam ser varridas para atender uma consulta.

Pré-requisitos:

Antes de criar um índice de grade espacial:

- Seu ID do usuário deve conter as autorizações requeridas pela instrução DB2 SQL CREATE INDEX. O ID do usuário deve ter pelo menos uma das seguintes autoridades ou privilégios:
  - Autoridade SYSADM ou DBADM no banco de dados no qual reside a tabela que contém a coluna
  - Duas das seguintes autoridades ou privilégios:
    - Um dos seguintes privilégios de tabela:
      - Privilégio CONTROL na tabela
      - Privilégio INDEX na tabela
    - Uma das seguintes autorizações ou privilégios no esquema:
      - Autoridade IMPLICIT_SCHEMA no banco de dados, se o esquema implícito ou explícito do índice não existir
      - Privilégio CREATEIN no esquema, se o nome do esquema do índice se referir a um esquema existente
- Você deve conhecer os valores que deseja especificar para o nome completo do índice de grade espacial e os três tamanhos de grades que o índice utilizará. Para obter informações adicionais, consulte "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113.

Recomendações:

- Antes de criar um índice de grade espacial em uma coluna, utilize o Orientador de Índice para determinar os parâmetros para o índice. O Orientador de Índice pode analisar os dados da coluna espacial e sugerir tamanhos de grades apropriados para seu índice de grade espacial.
- Se você planeja fazer um carregamento inicial de dados na coluna, deverá criar o índice de grade espacial depois de concluir o processo de carregamento. Dessa forma, você pode escolher os melhores tamanhos de células de grade que estão baseados nas características dos dados, utilizando o Orientador de Índice. Além disso, carregar os dados antes de criar o índice melhora o desempenho do processo de carregamento, pois o índice de grade espacial não precisará ser mantido durante o processo de carregamento.

Restrições:

As mesmas restrições para criar índices utilizando a instrução CREATE INDEX entram em vigor quando você cria um índice de grade espacial. Ou seja, a coluna na qual você cria um índice deve ser uma coluna de tabela básica, não uma coluna de exibição ou uma coluna de pseudônimo. O DB2 UDB resolverá aliases no processo.

Procedimento:

Você pode criar um índice de grade espacial de uma das seguintes formas:

- Utilize a janela do Spatial Extender do Centro de Controle do DB2.
- Utilize a instrução SQL CREATE INDEX com a extensão db2gse.spatial_index na cláusula EXTEND USING.
- Utilize uma ferramenta GIS que funciona com o DB2 Spatial Extender. Se você utilizar tal ferramenta para criar o índice, a ferramenta emitirá a instrução SQL CREATE INDEX apropriada.

Esta seção apresenta as etapas para os primeiros dois métodos. Para obter informações sobre como utilizar uma ferramenta GIS para criar um índice de grade espacial, consulte a documentação que acompanha essa ferramenta.

Para criar um índice de grade espacial utilizando o Centro de Controle:

1. No Centro de Controle, clique com o botão direito na tabela que tem a coluna espacial na qual você deseja colocar um índice de grade espacial e selecione **Spatial Extender→Índices Espaciais** no menu pop-up. A janela Índices Espaciais aparece.
2. Siga as instruções na ajuda on-line para a janela Índices Espaciais. Você pode exibir estas instruções clicando no botão de comando **Ajuda** na janela Índices Espaciais.

Para executar esta tarefa utilizando a instrução SQL CREATE INDEX:

1. Determine a instrução CREATE INDEX utilizando a cláusula EXTEND USING e a extensão de índice de grade db2gse.spatial_index.

   Por exemplo, a instrução a seguir cria o índice de grade espacial TERRIDX para a tabela BRANCHES que possui uma coluna espacial TERRITORY.

   ```
   CREATE INDEX terridx
      ON branches (territory)
      EXTEND USING db2gse.spatial_index (1.0, 10.0, 100.0)
   ```
2. Emita o comando CREATE INDEX no Editor de Comandos do DB2, na Janela de Comandos do DB2 ou no processador da linha de comandos do DB2.

Conceitos Relacionados:

- "Índices de Grade Espaciais" na página 102
- "Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade" na página 104
- "Ajustando Índices de Grade Espaciais com o Orientador de Índice—Visão Geral" na página 112

Tarefas Relacionadas:

- "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113
- "Analisando Estatísticas de Índice de Grade Espacial" na página 114

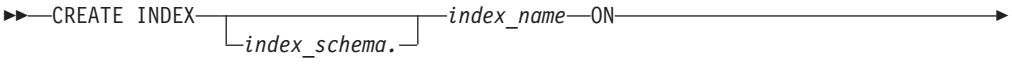
Referência Relacionada:

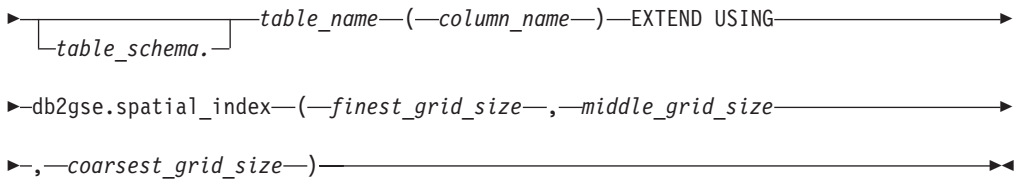
- "CREATE INDEX statement" na publicação *SQL Reference, Volume 2*
- "Instrução CREATE INDEX para um Índice de Grade Espacial" na página 110
- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Instrução CREATE INDEX para um Índice de Grade Espacial

Utilize a instrução CREATE INDEX com a cláusula EXTEND USING para criar um índice de grade espacial.

Sintaxe:





Parâmetros:

index_schema.
Nome do esquema ao qual deve pertencer o índice que está sendo criado. Se você não especificar um nome, o DB2 UDB utilizará o nome do esquema que está armazenado no registro especial CURRENT SCHEMA.

index_name
Nome não qualificado do índice da grade que está sendo criado.

table_schema.
Nome do esquema ao qual pertence a tabela que contém *column_name*. Se não for especificado um nome, o DB2 utilizará o nome do esquema que está armazenado no registro especial CURRENT SCHEMA.

table_name
Nome não qualificado da tabela que contém *column_name*.

column_name
Nome da coluna espacial na qual é criado o índice de grade espacial.

finest_grid_size, middle_grid_size, coarsest_grid_size
Tamanhos de grades para o índice de grade espacial. Estes parâmetros devem atender as seguintes condições:

- *finest_grid_size* deve ser maior do que 0.
- *middle_grid_size* deve ser maior que *finest_grid_size* ou deve ser 0.
- *coarsest_grid_size* deve ser maior que *middle_grid_size* ou deve ser 0.

Se você criar o índice de grade espacial utilizando o Centro de Controle ou a instrução CREATE INDEX, a validade dos tamanhos de grades será verificada quando a primeira geometria for indexada. Portanto, se os tamanhos de grade especificados não atenderem as condições de seus valores, ocorrerá uma condição de erro nos momentos descritos nestas situações:

- Se todas as geometrias na coluna espacial forem nulas, o Spatial Extender criará com êxito o índice, sem verificar a validade dos tamanhos de grades. O Spatial Extender valida os tamanhos de grades quando você insere ou atualiza uma geometria não nula nessa coluna espacial. Se os tamanhos de grades especificados não forem válidos, ocorrerá um erro quando você inserir ou atualizar a geometria não nula.
- Se existirem geometrias não nulas na coluna espacial enquanto você cria o índice, o Spatial Extender validará os tamanhos de grades neste momento. Se os tamanhos de grades especificados não forem válidos, ocorrerá um erro imediatamente e o índice de grade espacial não será criado.

Exemplos:

A instrução CREATE INDEX de exemplo a seguir cria o índice de grade espacial TERRIDX na coluna espacial TERRITORY na tabela BRANCHES:

```
CREATE INDEX terridx
   ON branches (territory)
   EXTEND USING db2gse.spatial_index (1.0, 10.0, 100.0)
```

Conceitos Relacionados:

- "Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade" na página 104
- "Índices de Grade Espaciais" na página 102
- "Ajustando Índices de Grade Espaciais com o Orientador de Índice—Visão Geral" na página 112

Tarefas Relacionadas:

- "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113
- "Analisando Estatísticas de Índice de Grade Espacial" na página 114
- "Criando Índices de Grades Espaciais" na página 108

Referência Relacionada:

- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Ajustando Índices de Grade Espaciais com o Orientador de Índice

Ajustando Índices de Grade Espaciais com o Orientador de Índice—Visão Geral

O DB2® Spatial Extender oferece um utilitário, chamado *Orientador de Índice*, que pode ser utilizado para:

- Determinar os tamanhos de grades apropriados para seus índices de grade espaciais.

  O Orientador de Índice analisa as geometrias em uma coluna espacial e recomenda os tamanhos de grades adequados para seu índice de grade espacial. Para o procedimento, consulte "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113.
- Analisar um índice de grade existente.

  O Orientador de Índice pode coletar e exibir estatísticas a partir das quais você pode determinar como os tamanhos de células de grade atuais facilitam a recuperação dos dados espaciais. Para o procedimento, consulte "Analisando Estatísticas de Índice de Grade Espacial" na página 114.

Conceitos Relacionados:

- "Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade" na página 104
- "Índices de Grade Espaciais" na página 102

Tarefas Relacionadas:

- "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113
- "Analisando Estatísticas de Índice de Grade Espacial" na página 114

Referência Relacionada:

- "Instrução CREATE INDEX para um Índice de Grade Espacial" na página 110
- "O Comando gseidx" na página 119
- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Determinando Tamanhos de Grades para um Índice de Grade Espacial

Antes de criar um índice de grade espacial em uma coluna, você pode utilizar o Orientador de Índice para determinar os tamanhos de grades apropriados.

Pré-requisitos:

Antes de analisar os dados que deseja indexar:

- Seu ID do usuário deve conter o privilégio SELECT nesta tabela.
- Se sua tabela tiver mais de um milhão de linhas, talvez você queira utilizar a cláusula ANALYZE para analisar um subconjunto das linhas para ter um tempo de processamento razoável. Você deve ter um espaço de tabelas USER TEMPORARY disponível para utilizar a cláusula ANALYZE. Defina o tamanho da página deste espaço de tabelas para pelo menos 8 KB e certifique-se de que tenha privilégios USE para ele. Por exemplo, as seguintes instruções DDL criam um conjunto de buffers com o mesmo tamanho de página que o espaço de tabelas temporário do usuário e concedem o privilégio USE a qualquer pessoa:

  ```
  CREATE BUFFERPOOL bp8k  SIZE 1000 PAGESIZE 8 K;
  CREATE USER TEMPORARY TABLESPACE usertempts
     PAGESIZE 8 K
     MANAGED BY SYSTEM USING ('c:\tempts')
     BUFFERPOOL bp8k
  GRANT USE OF TABLESPACE usertempts TO PUBLIC;
  ```

  Como opção, você pode utilizar o Centro de Controle do DB2 para criar um espaço de tabelas do usuário e o conjunto de buffers correspondente.

Procedimento:

Para determinar os tamanhos de grades apropriados para um índice de grade espacial:

1. Solicite ao Index Advisor para recomendar tamanhos de células para o índice que você deseja criar.
   a. Insira o comando que chama o Orientador de Índice com a palavra-chave ADVISE para solicitar tamanhos de células de grade. Por exemplo, para chamar o Orientador de Índice para a coluna FORMATO na tabela MUNICÍPIOS, insira:

      ```
      gseidx CONNECT TO mydb USER userID USING password GET GEOMETRY
      STATISTICS FOR COLUMN userID.counties(shape) ADVISE
      ```

      Restrição: Se você inserir o comando **gseidx** acima a partir de um prompt do sistema operacional, digite o comando inteiro na mesma linha. Como alternativa, você pode executar comandos **gseidx** a partir de um arquivo CLP, que permite dividir o comando em várias linhas.

      O Orientador de Índice retorna os tamanhos de células de grade recomendados. Por exemplo, o comando **gseidx** acima com a palavra-chave ADVISE retorna os seguintes tamanhos de célula recomendados para a coluna FORMATO:

      ```
      Tamanho Jan. Consul. Tamanhos Grad. Sug.          Custo
      ----------------  -------------------------       ----
            0.1          0.7,       2.8,      14.0       2.7
            0.2          0.7,       2.8,      14.0       2.9
            0.5          1.4,       3.5,      14.0       3.5
              1          1.4,       3.5,      14.0       4.8
              2          1.4,       3.5,      14.0       8.2
      ```

```
   5          1.4,       3.5,      14.0         24
  10          2.8,       8.4,      21.0         66
  20          4.2,      14.7,      37.0        190
  50          7.0,      14.0,      70.0        900
 100         42.0,         0,         0       2800
```

b. Escolha um tamanho de janela de consulta apropriado a partir da saída **gseidx**, com base na largura das coordenadas exibidas em sua tela.

   Neste exemplo, os valores de latitude e longitude em graus decimais representam as coordenadas. Se sua exibição de mapa típica tiver uma largura de aproximadamente 0,5 graus (aproximadamente 55 quilômetros), vá para a linha que possui o valor 0,5 na coluna Tamanho da Janela de Consulta. Esta linha sugeriu tamanhos de grade de 1.4, 3.5 e 14.0.

2. Crie o índice com os tamanhos de grade sugeridos. Por exemplo, na etapa anterior, você pode executar a seguinte instrução SQL:

```
CREATE INDEX counties_shape_idx ON userID.counties(shape)
EXTEND USING DB2GSE.SPATIAL_INDEX(1.4,3.5,14.0);
```

Conceitos Relacionados:

- "Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade" na página 104
- "Índices de Grade Espaciais" na página 102

Tarefas Relacionadas:

- "Analisando Estatísticas de Índice de Grade Espacial" na página 114

Referência Relacionada:

- "Instrução CREATE INDEX para um Índice de Grade Espacial" na página 110
- "O Comando gseidx" na página 119
- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Analisando Estatísticas de Índice de Grade Espacial

As estatísticas sobre um índice de grade espacial existente podem informar se o índice é eficiente ou se ele deve ser substituído por um índice mais eficiente. Utilize o Orientador de Índice para obter estas estatísticas e, se necessário, para substituir o índice.

Recomendação: Igualmente importante para ajustar seu índice é verificar se ele está sendo utilizado por suas consultas. Para determinar se um índice espacial está sendo utilizado, execute o Visual Explain no Centro de Controle do DB2 ou uma ferramenta da linha de comandos tal como **db2exfmt** em sua consulta. Na seção "Plano de Acesso" da saída de explicação, se você vir um operador EISCAN e o nome de seu índice espacial, a consulta utilizará seu índice.

Pré-requisitos:

Antes de analisar os dados que deseja indexar:

- Seu ID do usuário deve conter o privilégio SELECT nesta tabela.
- Se sua tabela tiver mais de um milhão de linhas, talvez você queira utilizar a cláusula ANALYZE para analisar um subconjunto das linhas para ter um tempo de processamento razoável. Você deve ter um espaço de tabelas USER TEMPORARY disponível para utilizar a cláusula ANALYZE. Defina o tamanho da página deste espaço de tabelas para pelo menos 8 KB e certifique-se de que tenha privilégios USE para ele. Por exemplo, as seguintes instruções DDL criam

um conjunto de buffers com o mesmo tamanho de página que o espaço de tabelas temporário do usuário e concedem o privilégio USE a qualquer pessoa:

```
CREATE BUFFERPOOL bp8k  SIZE 1000 PAGESIZE 8 K;
CREATE USER TEMPORARY TABLESPACE usertempts
   PAGESIZE 8 K
   MANAGED BY SYSTEM USING ('c:\tempts')
   BUFFERPOOL bp8k
GRANT USE OF TABLESPACE usertempts TO PUBLIC;
```

Como opção, você pode utilizar o DB2 Control Center para criar um espaço de tabelas do usuário e o conjunto de buffers correspondente.

Procedimento:

Para obter estatísticas sobre um índice de grade espacial e, se necessário, substituir o índice:

1. Deixe o Orientador de Índice coletar estatísticas com base nos tamanhos de células de grade do índice existente. Você pode solicitar estatísticas sobre um subconjunto de dados indexados ou sobre todos os dados.
   - Para obter estatísticas de dados indexados em um subconjunto de linhas, insira o comando **gseidx** e especifique a palavra-chave ANALYZE e seus parâmetros, além da cláusula de índice existente e da palavra-chave DETAIL. Você pode especificar o número ou a porcentagem de linhas que o Orientador de Índice deve analisar para obter estatísticas.Por exemplo, para obter estatísticas sobre um subconjunto dos dados indexados pelo índice COUNTIES_SHAPE_IDX, insira:

     ```
     gseidx CONNECT TO mydb USER userID USING password GET GEOMETRY
     STATISTICS FOR INDEX userID.counties_shape_idx DETAIL ANALYZE 25 PERCENT
      ADVISE
     ```

   - Para obter estatísticas sobre todos os dados indexados, insira o comando **gseidx** e especifique sua cláusula de índice existente. Inclua a palavra-chave DETAIL. Por exemplo, para chamar o Orientador de Índice para o índice COUNTIES_SHAPE_IDX, insira:

     ```
     gseidx CONNECT TO mydb USER userID USING password GET GEOMETRY
     STATISTICS FOR INDEX userID.counties_shape_idx DETAIL SHOW HISTOGRAM ADVISE
     ```

   O Orientador de Índice retorna estatísticas, um histograma dos dados e tamanhos de células recomendados para o índice existente. Por exemplo, o comando **gseidx** acima para todos os dados indexados por COUNTIES_SHAPE_IDX retorna as seguintes estatísticas:

   ```
   Nível de Grade 1
   ------------

   Tamanho da Grade              : 0.5
   Número de Geometrias          : 2936
   Número de Entradas de Índice  : 12197

   Número de Células de Grade Ocupadas       : 2922
   Proporção Entrada de Índice/Geometria     : 4.154292
   Proporção Geometria/Célula da Grade       : 1.004791
   Número máximo de Geometrias por Célula da Grade : 14
   Número mínimo de Geometrias por Células da Grade:  1

   Entradas de Índice : 1     2     3      4      10
   ---------------  ------ ------ ------ ------ ------
   Absoluto      :  86     564    72     1519   695
   Porcentagem (%): 2.93   19.21  2.45   51.74  23.67
   ```

```
Nível de Grade 2
------------

Tamanho da Grade              : 0.0
Não existem geometrias indexadas neste nível.


Nível de Grade 3
------------

Tamanho da Grade              : 0.0
Não existem geometrias indexadas neste nível.


Nível de Grade X
------------

Número de Geometrias          : 205
Número de Entradas de Índice  : 205
```

2. Determine como os tamanhos de células de grade do índice existente facilitarão a recuperação. Avalie as estatísticas retornadas na etapa anterior.

   Dicas:

   - A estatística "Proporção Entrada de Índice/Geometria" deve ser um valor no intervalo de 1 a 4, preferencialmente valores mais próximos de 1.
   - O número de entradas de índice por geometria deve ser menor que 10 no maior tamanho de grade para evitar o nível de estouro.

     A aparência da seção "Nível de Grade X" na saída do Orientador de Índice indica que existe um nível de estouro.

   As estatísticas de índice obtidas na etapa anterior para COUNTIES_SHAPE_IDX indicam que os tamanhos de grade (0.5, 0, 0) não são apropriados para os dados nesta coluna porque:

   - Para o Nível de Grade 1, o valor da "Proporção Entrada de Índice/Geometria" `4.154292` é maior que a diretriz de 4.

     A linha de "Entradas de Índice" possui os valores 1, 2, 3, 4 e 10, que indica o número de entradas de índice por geometria. Os valores "Absolutos" abaixo de cada coluna de "Entradas de Índice" indicam o número de geometrias que possuem um número específico de entradas de índice. Por exemplo, a saída na etapa anterior mostra 1519 geometrias que possuem 4 entradas de índice. O valor "Absoluto" para 10 entradas de índice é 695, que indica que 695 geometrias possuem entre 5 e 10 entradas de índice.
   - A aparência da seção "Nível de Grade X" indica que existe um nível de índice de estouro. As estatísticas mostram que 205 geometrias possuem mais de 10 entradas de índice cada.

3. Se as estatísticas não forem satisfatórias, consulte a seção "Histograma" e as linhas apropriadas nas colunas "Tamanho da Janela de Consulta" e "Tamanhos de Grades Sugeridos" na saída do Orientador de Índice:

   a. Localize o tamanho do MBR com o maior número de geometrias. A seção "Histograma" lista os tamanhos de MBR e o número de geometrias que possuem esse tamanho de MBR. No histograma de amostra a seguir, o maior número de geometrias (437) está no tamanho de MBR 0.5.

      ```
      Histograma:
      ----------
          Tamanho MBR          Contagem Geométrica
          -------------------- --------------------
                     0.040000                    1
                     0.045000                    3
                     0.050000                    1
      ```

```
                0.055000                3
                0.060000                3
                0.070000                4
                0.075000                3
                0.080000                4
                0.085000                1
                0.090000                2
                0.095000                1
                0.150000               10
                0.200000                9
                0.250000               15
                0.300000               23
                0.350000               83
                0.400000              156
                0.450000              282
                0.500000              437
                0.550000              397
                0.600000              341
                0.650000              246
                0.700000              201
                0.750000              154
                0.800000              120
                0.850000               66
                0.900000               79
                0.950000               59
                1.000000               47
                1.500000              230
                2.000000               89
                2.500000               34
                3.000000               10
                3.500000                5
                4.000000                3
                5.000000                3
                5.500000                2
                6.000000                2
                6.500000                3
                7.000000                2
                8.000000                1
               15.000000                3
               25.000000                2
               30.000000                1
```

b. Vá para a linha Tamanho da Janela de Consulta com o valor 0.5 para obter os tamanhos de grades sugeridos (1.4, 3.5, 14.0).

```
Tamanho Jan. Consul. Tamanhos Grad. Sug.           Custo
----------------  -------------------------        ----
       0.1          0.7,       2.8,      14.0       2.7
       0.2          0.7,       2.8,      14.0       2.9
       0.5          1.4,       3.5,      14.0       3.5
         1          1.4,       3.5,      14.0       4.8
         2          1.4,       3.5,      14.0       8.2
         5          1.4,       3.5,      14.0        24
        10          2.8,       8.4,      21.0        66
        20          4.2,      14.7,      37.0       190
        50          7.0,      14.0,      70.0       900
       100         42.0,         0,         0      2800
```

4. Verifique se os tamanhos recomendados atendem as diretrizes na etapa 2. Execute o comando **gseidx** com os tamanhos de grades sugeridos:

```
gseidx CONNECT TO mydb USER userID USING  password GET GEOMETRY
 STATISTICS FOR COLUMN userID.counties(shape) USING GRID SIZES (1.4, 3.5, 14.0)

Nível de Grade 1
------------

Tamanho da Grade             : 1.4
Número de Geometrias         : 3065
Número de Entradas de Índice  : 5951
```

```
Número de Células de Grade Ocupadas             : 513
Proporção Entrada de Índice/Geometria           : 1.941599
Proporção Geometria/Célula da Grade             : 5.974659
Número máximo de Geometrias por Célula da Grade : 42
Número mínimo de Geometrias por Células da Grade:  1

Entradas Índice: 1      2      3      4      10
---------------  ------ ------ ------ ------ ------
Absoluto       : 1180   1377   15     493    0
Porcentagem (%): 38.50  44.93  0.49   16.08  0.00


Nível Grade 2
------------

Tamanho da Grade              : 3.5
Número de Geometrias          : 61
Número de Entradas de Índice  : 143

Número de Células de Grade Ocupadas             : 56
Proporção Entrada de Índice/Geometria           : 2.344262
Proporção Geometria/Célula da Grade             : 1.089286
Número máximo de Geometrias por Célula da Grade : 10
Número mínimo de Geometrias por Células da Grade:  1

Entradas Índice: 1      2      3      4      10
---------------  ------ ------ ------ ------ ------
Absoluto       : 15     28     0      18     0
Porcentagem (%): 24.59  45.90  0.00   29.51  0.00


Nível Grade 3
------------

Tamanho da Grade              : 14.0
Número de Geometrias          : 15
Número de Entradas de Índice  : 28

Número de Células de Grade Ocupadas             : 9
Proporção Entrada de Índice/Geometria           : 1.866667
Proporção Geometria/Célula da Grade             : 1.666667
Número máximo de Geometrias por Célula da Grade : 10
Número mínimo de Geometrias por Células da Grade: 1

Entradas Índice: 1      2      3      4      10
---------------  ------ ------ ------ ------ ------
Absoluto       :  7     5      1      2      0
Porcentagem (%): 46.67  33.33  6.67   13.33  0.00
```

As estatísticas agora mostram valores nas diretrizes:

- Os valores da "Proporção Entrada de Índice/Geometria" são 1.941599 para o Nível de Grade 1, 2.344262 para o Nível de Grade 2 e 1.866667 para o Nível de Grade 3. Estes valores estão no intervalo de valores de diretrizes de 1 a 4.
- A ausência da seção "Nível de Grade X" indica que não existe nenhuma entrada de índice no nível de estouro.

5. Elimine o índice existente e substitua-o por um índice que especifique os tamanhos de grades recomendados. Para a amostra na etapa anterior, execute as seguintes instruções DDL:

```
DROP INDEX userID.counties_shape_idx;
CREATE INDEX counties_shape_idx ON userID.counties(shape) EXTEND USING
  DB2GSE.SPATIAL_INDEX(1.4,3.5,14.0);
```

Conceitos Relacionados:

- "Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade" na página 104
- "Índices de Grade Espaciais" na página 102

Tarefas Relacionadas:

- "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113
- "Criando Índices de Grades Espaciais" na página 108

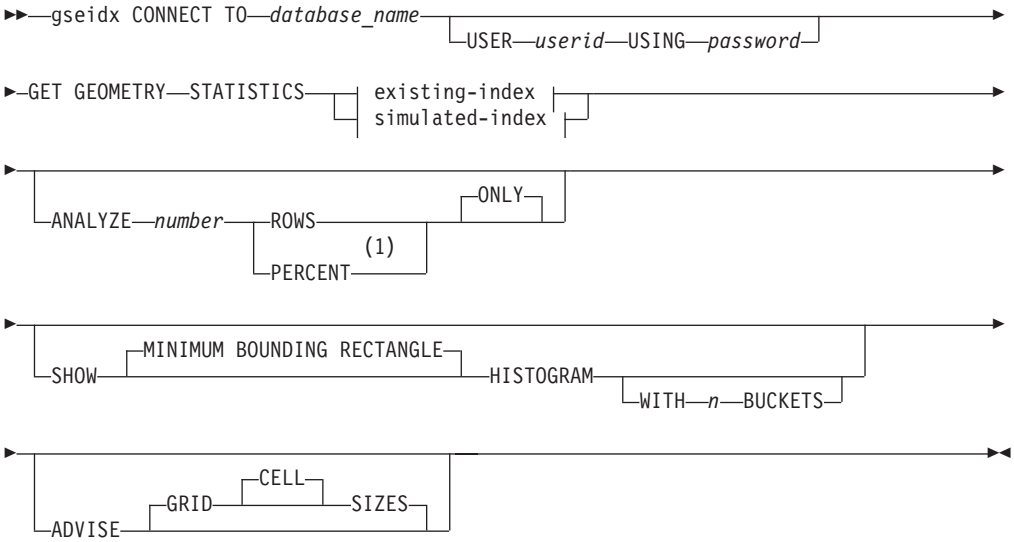
Referência Relacionada:

- "Instrução CREATE INDEX para um Índice de Grade Espacial" na página 110
- "O Comando gseidx" na página 119
- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

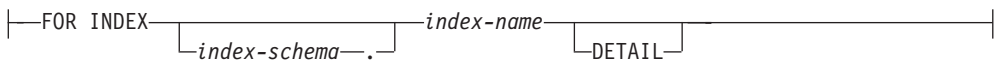
O Comando gseidx

Utilize o comando **gseidx** para chamar o Orientador de Índice para índices de grade espaciais.

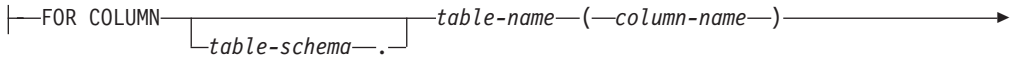
Sintaxe

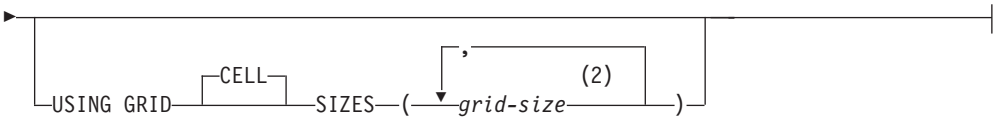


existing-index:



simulated-index:





Notas:

1 Em vez da palavra-chave PERCENT, você pode especificar um sinal de porcentagem (%).

2 Você pode especificar tamanhos de células para um, dois ou três níveis de grade.

Parâmetros:

database_name
O nome do banco de dados no qual a tabela espacial reside.

userid O ID do usuário que possui autoridade SYSADM ou DBADM no banco de dados no qual o índice ou a tabela reside ou a autoridade SELECT na tabela. Se você efetuar logon no ambiente de comandos do DB2 com o ID do usuário do proprietário do banco de dados, não será necessário especificar *userid* e *password* no comando **gseidx**.

password
Senha do ID do usuário.

existing-index:

Refere-se a um índice existente do qual serão coletadas estatísticas.

index-schema
Nome do esquema que inclui o índice existente.

index-name
Nome não qualificado do índice existente.

DETAIL
Mostra as seguintes informações sobre cada nível de grade:

- O tamanho das células de grade
- O número de geometrias indexadas
- O número de entradas do índice
- O número de células de grade que contêm geometrias
- O número médio de entradas de índice por geometria
- O número médio de geometrias por célula de grade
- O número de geometrias na célula que contém a maior quantidade de geometrias
- O número de geometrias na célula que contém a menor quantidade de geometrias

simulated-index:

Refere-se a uma coluna de tabela e a um índice simulado para esta coluna.

table-schema
Nome do esquema que inclui a tabela com a coluna à qual se destina o índice simulado.

table-name
Nome não qualificado da tabela com a coluna à qual se destina o índice simulado.

column-name
Nome não qualificado da coluna da tabela à qual se destina o índice simulado.

grid-size
Tamanhos de células em cada nível de grade (nível mais fino, nível médio e nível mais espesso) de um índice simulado. Você deve especificar um tamanho de célula para pelo menos um nível. Se não desejar incluir um nível, não especifique um tamanho de célula de grade para ele ou especifique um tamanho de célula de grade zero (0.0) para ele.

Quando especificar o parâmetro *grid-size*, o Orientador de Índice retornará os mesmos tipos de estatísticas que ele retorna quando você inclui a palavra-chave DETAIL na cláusula existing-index.

ANALYZE *number* **ROWS | PERCENT ONLY**
Reúne estatísticas sobre dados em um subconjunto de linhas de tabela. Se sua tabela tiver mais de um milhão de linhas, convém utilizar a cláusula ANALYZE para ter um tempo de processamento razoável. Especifique a quantidade aproximada ou a porcentagem aproximada das linhas a serem incluídas neste subconjunto.

SHOW MINIMUM BOUNDING RECTANGLE HISTOGRAM
Exibe um gráfico que mostra os tamanhos dos MBRs (Minimum Bounding Rectangles) das geometrias e o número de geometrias cujos MBRs têm o mesmo tamanho.

WITH *n* **BUCKETS**
Especifica o número de agrupamentos para os MBRs de todas as geometrias analisadas. Os MBRs pequenos são agrupados com outras geometrias pequenas. Os MBRs grandes são agrupados com outras geometrias grandes.

Se você não especificar este parâmetro ou especificar 0 reservatórios, o Orientador de Índice exibirá tamanhos de reservatórios logarítmicos. Por exemplo, o tamanho do MBR pode ser de valores logarítmicos como 1.0, 2.0, 3.0,... 10.0, 20.0, 30.0,... 100.0, 200.0, 300.0,...

Se você especificar um número de reservatórios maior que 0, o Orientador de Índice exibirá valores de mesmo tamanho. Por exemplo, os tamanhos de MBR podem ter valores de mesmo tamanho como 8.0, 16.0, 24.0,... 320.0, 328.0, 334.0.

O padrão é utilizar reservatórios de tamanhos logarítmicos.

ADVISE GRID CELL SIZES
Calcula tamanhos de células de grade próximos do ideal.

Nota de Uso:

Se você inserir o comando **gseidx** a partir de um prompt do sistema operacional, será necessário digitar todo o comando em uma única linha.

Exemplo:

O exemplo a seguir é um pedido para retornar informações detalhadas sobre um índice de grade existente cujo nome é COUNTIES_SHAPE_IDX e para sugerir tamanhos de índice de grade apropriados:

```
gseidx CONNECT TO mydb USER user ID USING password GET GEOMETRY
STATISTICS FOR INDEX userID.counties_shape_idx DETAIL ADVISE
```

Para obter uma explicação das informações que este comando retorna, consulte "Analisando Estatísticas de Índice de Grade Espacial" na página 114.

Conceitos Relacionados:

- "Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade" na página 104
- "Índices de Grade Espaciais" na página 102
- "Ajustando Índices de Grade Espaciais com o Orientador de Índice—Visão Geral" na página 112

Tarefas Relacionadas:

- "Determinando Tamanhos de Grades para um Índice de Grade Espacial" na página 113
- "Criando Índices de Grades Espaciais" na página 108

Referência Relacionada:

- "Instrução CREATE INDEX para um Índice de Grade Espacial" na página 110
- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Utilizando Exibições para Acessar Colunas Espaciais

Você pode definir uma exibição que utiliza uma coluna espacial da mesma forma que define exibições no DB2 para outros tipos de dados. Se você tiver uma tabela com uma coluna espacial e deseja que uma exibição utilize-a, utilize as seguintes fontes de informações.

Tarefas Relacionadas:

- "Creating a view" na publicação *Administration Guide: Implementation*

Referência Relacionada:

- "CREATE VIEW statement" na publicação *SQL Reference, Volume 2*

Capítulo 12. Analisando e Gerando Informações Espaciais

Depois de ocupar colunas espaciais, você está pronto para consultá-las. Este capítulo:

- Descreve os ambientes nos quais você pode enviar consultas
- Fornece exemplos dos vários tipos de funções espaciais que podem ser chamadas em uma consulta
- Fornece diretrizes sobre como utilizar funções espaciais junto com índices espaciais

Ambientes para Executar a Análise Espacial

Você pode executar análise espacial utilizando SQL e funções espaciais nos seguintes ambientes de programação:

- Instruções SQL interativas.

  Você pode inserir instruções SQL interativas a partir do Editor de Comandos do DB2®, da Janela de Comandos do DB2 ou do processador da linha de comandos do DB2.
- Programas aplicativos em todas as linguagens suportadas pelo DB2.

Exemplos de Como as Funções Espaciais Operam

O DB2 Spatial Extender fornece funções que executam várias operações em dados espaciais. Geralmente, estas funções podem ser classificadas de acordo com o tipo de operação que elas executam. A Tabela 5 lista estas categorias, junto com exemplos. O texto após a Tabela 5 mostra a codificação destes exemplos.

Tabela 5. Operações e Funções Espaciais

Categoria de Função	Exemplo de Operação
Retorna informações sobre geometrias específicas.	Retorna a extensão, em milhas quadradas, da área de vendas da Loja 10.
Faz comparações.	Determina se a localização da residência de um cliente está dentro da área de vendas da Loja 10.
Gera novas geometrias a partir de existentes.	Gera a área de vendas de uma loja a partir de sua localização.
Converte geometrias em e a partir de formatos de troca de dados.	Converte informações do cliente em formato GML em uma geometria para que as informações possam ser incluídas em um banco de dados DB2.

Exemplo 1: Retorna informações sobre geometrias específicas:

Neste exemplo, a função ST_Area retorna um valor numérico que representa a área de vendas da loja 10. A função retornará a área nas mesmas unidades que as unidades do sistema de coordenadas que está sendo utilizado para definir a localização da área.

```
SELECT db2gse.ST_Area(sales_area)
FROM   stores
WHERE  id = 10
```

O exemplo a seguir mostra a mesma operação que o anterior, mas com ST_Area chamado como um método e retornando a área em unidades de milhas quadradas.

```
SELECT saleas_area..ST_Area('STATUTE MILE')
FROM   stores
WHERE  id = 10
```

Exemplo 2: Faz comparações:

Neste exemplo, a função ST_Within compara as coordenadas da geometria representando a residência de um cliente com as coordenadas de uma geometria representando a área de vendas da loja 10. A saída da função definirá se a residência está localizada na área de vendas.

```
SELECT c.first_name, c.last_name, db2gse.ST_Within(c.location, s.sales_area)
FROM   customers as c. stores AS s
WHERE  s.id = 10
```

Exemplo 3: Origina novas geometrias a partir das existentes.:

Neste exemplo, a função ST_Buffer gera uma geometria representando uma área de vendas de uma loja a partir de uma geometria representando a localização da loja.

```
UPDATE stores
SET    sales_area = db2gse.ST_Buffer(location, 10, 'KILOMETERS')
WHERE  id = 10
```

O exemplo a seguir mostra a mesma operação que o anterior, mas com ST_Buffer chamado como um método.

```
UPDATE stores
SET    sales_area = location..ST_Buffer(10, 'KILOMETERS')
WHERE  id = 10
```

Exemplo 4: Converte geometrias em e a partir de formatos de troca de dados.:

Neste exemplo, as informações do cliente codificadas em GML são convertidas em uma geometria, para que possam ser armazenadas em um banco de dados DB2.

```
INSERT
INTO   c.name,c.phoneNo,c.address
VALUES ( 123, 'Mary Anne', Smith', db2gse.ST_Point('
<gml:Point><gml:coord><gml=X>-130.876</gml:X>
<gml:Y>41.120'</gml:Y></gml:coord></gml:Point>, 1) )
```

Funções que Utilizam Índices para Otimizar Consultas

Um grupo especializado de funções espaciais, chamado de *funções de comparação*, pode aprimorar o desempenho da consulta, explorando um índice de grade espacial ou um índice geodésico de Voronoi (ambos conhecidos como *índices espaciais*). Cada uma destas funções compara duas geometrias. Se os resultados da comparação atenderem alguns critérios, a função retornará um valor 1; se os resultados não atenderem os critérios, a função retornará um valor 0. Se a comparação não puder ser executada, a função poderá retornar um valor nulo.

Por exemplo, a função ST_Overlaps compara duas geometrias que têm a mesma dimensão (por exemplo, duas cadeias de linhas ou dois polígonos). Se as geometrias forem parcialmente sobrepostas e se o espaço coberto pela sobreposição tiver a mesma dimensão das geometrias, ST_Overlaps retornará um valor 1.

A Tabela 6 na página 125 mostra quais funções de comparação podem utilizar um índice de grade espacial e quais podem utilizar um índice geodésico de Voronoi:

Tabela 6. Funções de Comparação que Podem Utilizar um Índice de Grade Espacial ou um Índice Geodésico de Voronoi

Função de Comparação	Pode Utilizar Índice de Grade Espacial	Pode Utilizar Índice Geodésico de Voronoi
EnvelopesIntersect	Sim	Sim
ST_Contains	Sim	Sim
ST_Crosses	Sim	Não
ST_Distance	Sim	Sim
ST_EnvIntersects	Sim	Sim
ST_Equals	Sim	Não
ST_Intersects	Sim	Sim
ST_MBRIntersects	Sim	Sim
ST_Overlaps	Sim	Não
ST_Touches	Sim	Não
ST_Within	Sim	Sim

Devido ao tempo e a memória requeridos para executar uma função, tal execução pode envolver um processamento considerável. Além disso, quanto mais complexas forem as geometrias que estão sendo comparadas, mais complexa e mais demorada será a comparação. As funções especializadas listadas acima podem concluir suas operações mais rapidamente se puderem utilizar um índice espacial para localizar geometrias. Para ativar tal função para utilizar um índice espacial, observe todas as seguintes regras:

- A função deve ser especificada em uma cláusula WHERE. Se for especificada em uma cláusula SELECT, HAVING ou GROUP BY, não será possível utilizar um índice espacial.
- A função deve ser a expressão à esquerda do predicado.
- O operador que é utilizado no predicado que compara o resultado da função com outra expressão deve ser um sinal de igual, com uma exceção: a função ST_Distance deve utilizar o operador menor que.
- A expressão à direita do predicado deve ser a constante 1, exceto quando ST_Distance é a função à esquerda.
- A operação deve envolver uma pesquisa em uma coluna espacial na qual um índice espacial está definido.

Por exemplo:

```
SELECT   c.name, c.address, c.phone
FROM customers AS c, bank_branches AS b
WHERE db2gse.ST_Distance(c.location, b.location) < 10000
  and b.branch_id = 3
```

A Tabela 7 na página 126 mostra as formas correta e incorreta de criar consultas espaciais para utilizar um índice espacial.

Tabela 7. Demonstração de Como as Funções Espaciais Podem Estar de Acordo e Violar Regras para Utilizar um Índice Espacial.

Consultas que Se Referem a Funções Espaciais	Regras Violadas

`SELECT *`
`FROM stores AS s`
`WHERE db2gse.ST_Contains(s.sales_zone,`
`ST_Point(-121.8,37.3, 1)) = 1`
	Nenhuma condição é violada neste exemplo.

`SELECT *`
`FROM stores AS s`
`WHERE db2gse.ST_Length(s.location) > 10`
	A função espacial ST_Length não compara geometrias e não pode utilizar um índice espacial.

`SELECT *`
`FROM stores AS s`
`WHERE 1=db2gse.ST_Within(s.location,:BayArea)`
	A função deve ser uma expressão à esquerda do predicado.

`SELECT *`
`FROM stores AS s`
`WHERE db2gse.ST_Contains(s.sales_zone,`
`ST_Point(-121.8,37.3, 1)) <> 0`
	Comparações de igualdade devem usar o inteiro constante 1.

`SELECT *`
`FROM stores AS s`
`WHERE db2gse.ST_Contains(ST_Polygon`
`('polygon((10 10, 10 20, 20 20, 20 10, 10 10))', 1),`
`ST_Point(-121.8, 37.3, 1) = 1`
	Não existe nenhum índice espacial em nenhum dos argumentos da função, portanto, nenhum índice poderá ser utilizado.

Conceitos Relacionados:

- "Considerações sobre o Número de Níveis de Índices e Tamanhos de Grade" na página 104
- "Índices Geodésicos de Voronoi" na página 177
- "Índices de Grade Espaciais" na página 102
- "Ajustando Índices de Grade Espaciais com o Orientador de Índice—Visão Geral" na página 112

Tarefas Relacionadas:

- "Criando Índices Geodésicos de Voronoi" na página 181
- "Criando Índices de Grades Espaciais" na página 108

Capítulo 13. Comandos do DB2 Spatial Extender

Este capítulo explica os comandos utilizados para configurar o DB2 Spatial Extender. Também explica como utilizar estes comandos para implementar projetos.

Chamando Comandos para Configurar o DB2 Spatial Extender e Desenvolvendo Projetos

Você pode utilizar um CLP (Processador da Linha de Comandos) chamado db2se, para configurar o Spatial Extender e criar projetos que utilizem dados espaciais. Este tópico explica como utilizar o db2se para executar comandos do DB2 Spatial Extender.

Pré-requisitos:

Antes de emitir comandos do db2se, você deve ter autorização para isso. Para saber qual autorização é necessária para um determinado comando, consulte a Tabela 8 para o tópico de procedimento armazenado associado ao comando. Por exemplo, o comando **db2se create_srs** exige as mesmas autoridades que o procedimento armazenado db2.ST_create_srs.

Exceção: O comando **db2se shape_info** não chama um procedimento armazenado. Em vez disso, ele exibe informações sobre o conteúdo de arquivos de formas.

Procedimento:

Digite comandos do db2se a partir de um prompt do sistema operacional.

Para saber quais subcomandos e parâmetros podem ser especificados:

- Digite `db2se` ou `db2se -h` e pressione Enter. É exibida uma lista de subcomandos do db2se.
- Digite `db2se` e um subcomando ou `db2se` e um subcomando, seguidos de `-h`. Em seguida, pressione Enter. É exibida a sintaxe necessária para o subcomando. Nesta sintaxe:
  - Cada parâmetro é precedido por um traço e seguido por um marcador para um valor de parâmetro.
  - Os parâmetros entre colchetes são opcionais. Os outros parâmetros são obrigatórios.

Importante: Para sua conveniência, a sintaxe do comando pode ser recuperada interativamente no monitor; não é necessário procurar a sintaxe em outro lugar.

Para emitir um comando do db2se, digite `db2se`. Em seguida, digite um subcomando, seguido pelos parâmetros e valores de parâmetros requeridos pelo subcomando. Por último, pressione Enter.

Nas versões anteriores, todos os subcomandos do Spatial Extender precisavam ser precedidos por `gseadm` em vez de `db2se`. Quaisquer scripts gseadm criados nas

versões anteriores ainda funcionam na Versão 8.1, mas a IBM recomenda que seus scripts sejam migrados para utilizar o processador da linha de comandos db2se.

Esteja ciente de que:

- É necessário digitar o ID do usuário e senha que lhe concedem acesso ao banco de dados que acabou de ser especificado. Por exemplo, digite o ID e a senha se desejar conectar-se ao banco de dados como um usuário que não seja você mesmo. Sempre preceda o ID com o parâmetro `userId` e preceda a senha com o parâmetro `pw`.

  Se você não especificar um ID do usuário e senha, seu ID do usuário e senha atuais serão utilizados por padrão.
- Por padrão, os valores inseridos não fazem distinção entre maiúsculas e minúsculas. Para que eles façam distinção entre maiúsculas e minúsculas, coloque-os entre aspas duplas. Por exemplo, para especificar o nome de tabela mytable em minúsculas, digite ″mytable″

  Nota: pode ser necessário utilizar escape nas aspas, para assegurar que não sejam interpretadas pelo prompt do sistema (shell), por exemplo, especifique o seguinte:

  ```
  \"mytable\"
  ```

  Se um valor que faz distinção entre maiúsculas e minúsculas for qualificado por outro valor que faz distinção entre maiúsculas e minúsculas, delimite os dois valores individualmente; por exemplo:

  ```
  "myschema"."mytable"
  ```

  Coloque as cadeias entre aspas duplas; por exemplo:

  ```
  "select * from newtable"
  ```

Quando o comando db2se for executado, o procedimento armazenado que corresponde ao comando será chamado e a operação solicitada será executada.

Visão Geral dos Comandos do db2se:

A tabela a seguir indica quais comandos do db2se devem ser emitidos para executar as tarefas envolvidas na configuração do Spatial Extender e na criação de projetos que utilizem dados espaciais. Esta tabela também fornece exemplos de comandos do db2se e fornece informações sobre autorizações e parâmetros específicos de comandos. À direita da tarefa, na segunda coluna, você verá um link ou referência a informações sobre um procedimento armazenado. Este procedimento armazenado é chamado quando o comando é emitido. A autorização para utilizar o procedimento armazenado é igual à autorização para utilizar o comando; além disso, o comando e o procedimento armazenado compartilham os mesmos parâmetros. Para informações adicionais sobre autorização e os significados dos parâmetros, consulte a seção identificada pela referência.

Tabela 8. Comandos do db2se Ordenados Por Tarefa

Tarefa	Comando e Exemplo
Criar um sistema de coordenadas.	**db2se create_cs** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_create_coordsys. O exemplo a seguir cria um sistema de coordenadas denominado "mycoordsys". `db2se create_cs mydb -coordsysName \"mycoordsys\" -definition GEOCS[\"GCS_NORTH_AMERICAN_1983\", DATUM[\"D_North_American_1983\", SPHEROID[\"GRS_1980\",6387137,298.257222101]], PRIMEM[\"Greenwich\",0],UNIT[\"Degree\", 0.01745329251994329 55]]`
Criar um sistema de referência espacial.	**db2se create_srs** Os parâmetros específicos do comando são iguais aos do procedimento armazenado db2gse.ST_create_srs. Não é necessária nenhuma autorização. O exemplo a seguir cria um sistema de referência espacial denominado "mysrs". `db2se create_srs mydb -srsName \"mysrs\" -srsID 100 -xScale 10 -coordsysName \"GCS_North_American_1983\"`
Eliminar um sistema de referência espacial.	**db2se drop_srs** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_drop_srsdb2gse.ST_drop_srs. O exemplo a seguir elimina um sistema de referência espacial denominado "mysrs". `db2se drop_srs mydb -srsName \"mysrs\"`
Excluir uma definição do sistema de coordenadas.	**db2se drop_cs** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_drop_coordsysdb2gse.ST_drop_coordsys. O exemplo a seguir elimina um sistema de coordenadas denominado "mycoordsys". `db2se drop_cs mydb -coordsysName \"mycoordsys\"`
Desativar automaticamente uma configuração de dados de geocode.	**db2se disable_autogc** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_disable_autogeocoding. O exemplo a seguir desativa a codificação geográfica automática de uma coluna com codificação geográfica denominada MYCOLUMN na tabela MYTABLE. `db2se disable_autogc mydb -tableName \"mytable\" -columnName \"mycolumn\"`

Tabela 8. Comandos do db2se Ordenados Por Tarefa (continuação)

Tarefa	Comando e Exemplo
Ativar um banco de dados para operações espaciais.	**db2se enable_db** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_enable_dbdb2gse.ST_enable_db. O exemplo a seguir ativa um banco de dados chamado MYDB para operações espaciais. `db2se enable_db mydb`

Exportar dados para um arquivo de transferência SDE.	**db2se export_sde**
	
	Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.GSE_export_sdedb2gse.GSE_export_sde.
	
	O exemplo a seguir exporta dados da tabela MYSDETABLE, que contém a coluna espacial MYSPATIALCOLUMN, para um arquivo de transferência SDE denominado mysdefile.
	`db2se export_sde mydb -tableName \"mySDEtable\" -columnName \"mySpatialcolumn\" -fileName /home/myaccount/mysdefile`
	
	O próximo exemplo exporta dados de uma tabela denominada SPATIALTABLE para um arquivo SDE denominado sdex, que será criado no cliente DB2. Erros e mensagens informativas (por exemplo, horário de início e término da exportação e quantas linhas foram exportadas) são gravados em um arquivo denominado sdex.export.log.
	`db2se export_sde mydb -client -fileName sdex -selectStatement "SELECT * FROM spatialTable" -messagesFile sdex.export.log`

Exportar dados para arquivos de formas.	**db2se export_shape** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_export_shape. procedure. O exemplo a seguir exporta uma coluna espacial denominada MYCOLUMN e sua tabela associada, MYTABLE, para um arquivo shape denominado myshapefile. `db2se export_shape mydb -fileName /home/myaccount/myshapefile -selectStatement "select * from mytable"`

Tabela 8. Comandos do db2se Ordenados Por Tarefa (continuação)

Tarefa	Comando e Exemplo

Importar um arquivo de transferência SDE.	**db2se import_sde**
	
	Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.GSE_import_sdedb2gse.GSE_import_sde.
	
	O exemplo a seguir importa um arquivo de transferência SDE denominado mysdefile para a tabela MYSDETABLE, que contém uma coluna espacial denominada MYSPATIALCOLUMN. Deve ser emitida uma consolidação para cada dez registros.
	`db2se import_sde mydb -tableName \"mysdetable\" -columnName \"mySpatialcolumn\" -fileName /home/myaccount/"mysdefile" -commitScope 10`
	
	O próximo exemplo mostra como importar um arquivo SDE denominado sdex, que reside no cliente DB2. Neste exemplo, os dados são importados para uma tabela denominada SDETABLE (para uma coluna denominada ID) e uma consolidação é emitida a cada 100 registros. Os erros são gravados em um arquivo denominado sdex.exceptions.
	`db2se import_sde mydb -client -filename sdex -srsId 1234 -tableName sdeTable -idColumn id -commitScope 100 -messagesFile sdex.exceptions`

Importar arquivos de formas.	**db2se import_shape** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_import_shape. O comando a seguir importa um arquivo shape denominado myfile para uma tabela denominada MYTABLE. Durante a importação, os dados espaciais em myfile são inseridos em uma coluna de MYTABLE denominada MYCOLUMN. `db2se import_shape mydb -fileName \"myfile\" -srsName NAD83_SRS_1 -tableName \"mytable\" -spatialColumnName \"mycolumn\"`
Registrar um geocoder.	**db2se register_gc** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_register_geocoder. O exemplo a seguir registra um geocoder denominado “mygeocoder”, implementado por uma função denominada “myschema.myfunction”. `db2se register_gc mydb -geocoderName \"mygeocoder"\ -functionSchema \"myschema\" -functionName \"myfnction\" -defaultParameterValues "1, 'string',,cast(null as varchar(50))" -vendor myvendor -description "myvendor geocoder returning well-known text"`

Tabela 8. Comandos do db2se Ordenados Por Tarefa (continuação)

Tarefa	Comando e Exemplo
Registrar uma coluna espacial.	**db2se register_spatial_column** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_register_spatial_column. O exemplo a seguir registra uma coluna espacial denominada MYCOLUMN na tabela MYTABLE, com o sistema de referência espacial "USA_SRS_1". `db2se register_spatial_column mydb -tableName \"mytable\" -columnName \"mycolumn\" -srsName USA_SRS_1`
Remover os recursos que ativam um banco de dados para operações espaciais.	**db2se disable_db** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_disable_dbdb2gse.ST_disable_db. O exemplo a seguir remove os recursos que ativam o banco de dados MYDB para operações espaciais. `db2se disable_db mydb`
Remover uma configuração de operações de codificação geográfica.	**db2se remove_gc_setup** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_remove_gc_setup. O exemplo a seguir remove uma configuração para operações de codificação geográfica que se aplicam a uma coluna espacial denominada MYCOLUMN na tabela MYTABLE. `db2se remove_geocoding_setup mydb -tableName \"mytable\" -columnName \"mycolumn\"`
Executar um geocoder no modo batch.	**db2se run_gc** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_run_gc. O exemplo a seguir executa um geocoder no modo batch para ocupar uma coluna denominada MYCOLUMN em uma tabela denominada MYTABLE. `db2se run_gc mydb -tableName \"mytable\" -columnName \"mycolumn\"`
Configurar um geocoder para execução automática.	**db2se enable_autogeocoding** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_enable_autogeocoding. O exemplo a seguir configura a codificação geográfica automática de uma coluna denominada MYCOLUMN na tabela MYTABLE. `db2se enable_autogeocoding mydb -tableName \"mytable\" -columnName \"mycolumn\"`

Tabela 8. Comandos do db2se Ordenados Por Tarefa (continuação)

Tarefa	Comando e Exemplo
Configurar operações de geocoding.	**db2se setup_gc** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_setup_geocoding. O exemplo a seguir configura operações de codificação geográfica para ocupar uma coluna espacial denominada MYCOLUMN na tabela MYTABLE. `db2se setup_gc mydb -tableName \"mytable\" -columnName \"mycolumn\" -geocoderName \"db2se_USA_GEOCODER\" -parameterValues "address,city,state,zip,2,90,70,20,1.1,'meter',4.." -autogeocodingColumns address,city,state,zip commitScope 10`

Mostrar informações sobre um arquivo shape e seu conteúdo.	**db2se shape_info**
	
	Para utilizar este comando, você deve:
	• Ter permissão para ler o arquivo ao qual o comando se refere.
	• Poder conectar-se ao banco de dados que contém esse arquivo (se utilizar o parâmetro *–database*, que especifica que o sistema procura no banco de dados indicado sistemas de coordenadas e sistemas de referência espacial compatíveis).
	
	O exemplo a seguir mostra informações sobre um arquivo shape denominado myfile, localizado no diretório atual.
	`db2se shape_info -fileName myfile`
	
	O exemplo a seguir mostra informações sobre um arquivo shape de amostra do UNIX denominado offices. O parâmetro *–database* localiza todos os sistemas de coordenadas e de referência espacial compatíveis no banco de dados indicado (neste caso, MYDB).
	`db2se shape_info -fileName ~/sqllib/samples/spatial/data/offices -database myDB`

Mostrar informações sobre um arquivo SDE e seu conteúdo.	**db2se sde_info**
	
	Para utilizar este comando, você deve:
	• Ter permissão para ler o arquivo ao qual o comando se refere.
	• Poder conectar-se ao banco de dados que contém esse arquivo (se utilizar o parâmetro *–database*, que especifica que o sistema procura no banco de dados indicado sistemas de coordenadas e sistemas de referência espacial compatíveis).
	
	O exemplo a seguir mostra informações sobre um arquivo SDE denominado sdefile, localizado no diretório atual.
	`db2se sde_info -fileName myfile`
	
	O próximo exemplo mostra informações sobre um arquivo SDE denominado sdex e pesquisa no banco de dados denominado MYDB todos os sistemas de coordenadas e sistemas de referência espacial compatíveis.
	`db2se sde_info -fileName data/sdex -database myDB`

Tabela 8. Comandos do db2se Ordenados Por Tarefa (continuação)

Tarefa	Comando e Exemplo
Cancelar registro de um geocoder.	**db2se unregister_gc** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_unregister_geocoder. O exemplo a seguir cancela o registro de um geocoder denominado "mygeocoder". `db2se unregister_gc mydb -geocoderName \"mygeoco der\"`
Cancelar registro de uma coluna espacial.	**db2se unregister_spatial_column** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_unregister_spatial_column. O exemplo a seguir cancela registros de uma coluna espacial denominada MYCOLUMN na tabela MYTABLE. `db2se unregister_spatial_column mydb -tableName \"mytable\" -columnName \"mycolumn\"`
Atualizar uma definição do sistema de coordenadas.	**db2se alter_cs** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_alter_coordsysdb2gse.ST_alter_coordsys. O exemplo a seguir atualiza a definição de um sistema de coordenadas denominado "mycoordsys" com um novo nome de organização. `db2se alter_cs mydb -coordsysName \"mycoordsys\" -organization myNeworganizationb -tableName \"mytable\"`
Atualizar uma definição do sistema de referência espacial.	**db2se alter_srs** Os parâmetros específicos de comandos e as autorizações necessárias são iguais aos do procedimento armazenado db2gse.ST_alter_srsdb2gse.ST_alter_srs. O exemplo a seguir altera um sistema de referência espacial denominado "mysrs" com xOffset e descrição diferentes. `db2se alter_srs mydb -srsName \"mysrs\" -xOffset 35 -description "Este é o meu sistema de referência espacial."`

Capítulo 14. Escrevendo Aplicativos e Utilizando o Programa de Amostra

Este capítulo explica como escrever aplicativos do Spatial Extender.

Gerando Aplicativos para o DB2 Spatial Extender

Se você pretende escrever programas aplicativos que chamam procedimentos armazenados ou funções do DB2 Spatial Extender, leia as informações sobre tarefas e referências a seguir.

Conceitos Relacionados:

- "O Programa de Amostra do DB2 Spatial Extender" na página 137

Tarefas Relacionadas:

- "Chamando Procedimentos Armazenados do DB2 Spatial Extender a Partir de um Aplicativo" na página 136
- "Incluindo o arquivo de cabeçalho do DB2 Spatial Extender em aplicativos espaciais" na página 135

Incluindo o arquivo de cabeçalho do DB2 Spatial Extender em aplicativos espaciais

O DB2 Spatial Extender fornece um arquivo de cabeçalho que define constantes que podem ser utilizadas com os procedimentos armazenados e funções do DB2 Spatial Extender.

Recomendação: Se estiver planejando chamar procedimentos armazenados ou funções do DB2 Spatial Extender a partir de programas C ou C++, inclua este arquivo de cabeçalho em seus aplicativos espaciais.

Procedimento:

Para certificar-se de que seus aplicativos do DB2 Spatial Extender possam utilizar as definições necessárias neste arquivo de cabeçalho:

1. Inclua o arquivo de cabeçalho do DB2 Spatial Extender em seu programa aplicativo. O arquivo de cabeçalho tem o seguinte nome:

   ```
   db2gse.h
   ```

   O arquivo de cabeçalho está localizado no diretório *db2path*/include, em que *db2path* é o diretório de instalação no qual o DB2 Universal Database está instalado.
2. Certifique-se de que o caminho do diretório include esteja especificado em seu `arquivo pronto` com a opção de compilação.

Se você for construir aplicativos Windows de 64 bits em um sistema Windows de 32 bits, altere o parâmetro DB2_LIBS no arquivo `samples/spatial/makefile.nt` para acomodar os aplicativos de 64 bits. As alterações necessárias estão realçadas a seguir:

```
DB2_LIBS = $(DB2_DIR)\lib\Win64\db2cli.lib $(DB2_DIR)\lib\Win64\db2api.lib
```

Chamando Procedimentos Armazenados do DB2 Spatial Extender a Partir de um Aplicativo

Os procedimentos armazenados do DB2 Spatial Extender são criados quando você ativa o banco de dados para operações espaciais. Se estiver planejando gerar programas aplicativos que chamam qualquer um dos procedimentos armazenados do DB2 Spatial Extender, utilize a instrução SQL CALL e especifique o nome do procedimento armazenado.

Procedimento:

Para chamar procedimentos armazenados do DB2 Spatial Extender, execute as seguintes ações:

- Para chamar o procedimento armazenado ST_enable_db, que ativa um banco de dados para operações espaciais, especifique o nome do procedimento armazenado da seguinte forma:

  ```
  CALL db2gse!ST_enable_db
  ```

  O `db2gse!` nesta chamada representa o nome da biblioteca do DB2 Spatial Extender. O procedimento armazenado ST_enable_db é o único no qual você precisa incluir um ponto de exclamação na chamada (ou seja, `db2gse!`).
- Para chamar qualquer outro procedimento armazenado do DB2 Spatial Extender, especifique o nome do procedimento armazenado no seguinte formato, em que `db2gse` é o nome do esquema para todos os procedimentos armazenados do DB2 Spatial Extender e *spatial_procedure_name* é o nome do procedimento armazenado:

  ```
  CALL db2gse.spatial_procedure_name
  ```

  Observe que não é incluído nenhum ponto de exclamação na chamada precedente.

  Os procedimentos armazenados do DB2 Spatial Extender são mostrados na tabela a seguir.

Tabela 9.

Procedimento Armazenado	Descrição
GSE_export_sde	Exporta uma coluna espacial e sua tabela associada para um arquivo de transferência SDE.
GSE_import_sde	Importa um arquivo de transferência SDE para um banco de dados.
ST_alter_coordsys	Atualiza um atributo de um sistema de coordenadas no banco de dados.
ST_alter_srs	Atualiza um atributo de um sistema de referência espacial no banco de dados.
ST_create_coordsys	Cria um sistema de coordenadas no banco de dados.
ST_create_srs	Cria um sistema de referência espacial no banco de dados.
ST_disable_autogeocoding	Especifica que o DB2 Spatial Extender deve parar a sincronização de uma coluna na qual foi efetuado geocode com suas colunas de geocoding associadas.

Tabela 9. (continuação)

Procedimento Armazenado	Descrição
ST_disable_db	Remove recursos que permitem que o DB2 Spatial Extender armazene dados espaciais e suporte operações que são executadas nestes dados.
ST_drop_coordsys	Exclui um sistema de coordenadas do banco de dados.
ST_drop_srs	Exclui um sistema de referência espacial do banco de dados.
ST_enable_autogeocoding	Especifica que o DB2 Spatial Extender deve sincronizar uma coluna na qual foi efetuado geocode com suas colunas de geocoding associadas.
ST_enable_db	Fornece a um banco de dados os recursos necessários para ele armazenar dados espaciais e suportar operações.
ST_export_shape	Exporta dados selecionados no banco de dados para um arquivo de formas.
ST_import_shape	Importa um arquivo de formas para um banco de dados.
ST_register_geocoder	Registra um geocoder diferente do DB2SE_USA_GEOCODER, que faz parte do produto DB2 Spatial Extender.
ST_register_spatial_column	Registra uma coluna espacial e associa a ela um sistema de referência espacial.
ST_remove_geocoding_setup	Remove todas as informações de configuração de geocoding da coluna na qual foi efetuado geocode.
ST_run_geocoding	Executa um geocoder no modo batch.
ST_setup_geocoding	Associa uma coluna na qual deve ser efetuado geocode a um geocoder e configura os valores de parâmetros de geocoding correspondentes.
ST_unregister_geocoder	Cancela o registro de um geocoder diferente do DB2SE_USA_GEOCODER.
ST_unregister_spatial_column	Remove o registro de uma coluna espacial.

O Programa de Amostra do DB2 Spatial Extender

O programa de amostra do DB2® Spatial Extender, `runGseDemo`, tem duas finalidades. Você pode utilizar o programa de amostra para se familiarizar com a programação de aplicativos para o DB2 Spatial Extender e você pode utilizar o programa para verificar a instalação do DB2 Spatial Extender. Consulte a seção "Tarefas Relacionadas" no final deste tópico para obter informações adicionais sobre como verificar a instalação do Spatial Extender.

- No UNIX®, você pode localizar o programa runGseDemo no seguinte caminho:
  `$HOME/sqllib/samples/spatial`

  em que `$HOME` é o diretório pessoal do proprietário da instância.

- No Windows®, você pode localizar o programa runGseDemo no seguinte caminho:

  ```
  c:\Arquivos de Programas\IBM\sqllib\samples\spatial
  ```

  em que `c:\Arquivos de Programas\IBM\sqllib` é o diretório no qual você instalou o DB2 Spatial Extender.

O programa de amostra do DB2 Spatial Extender `runGseDemo` facilita a programação de aplicativos. Utilizando este programa de amostra, você poderá ativar um banco de dados para operações espaciais e executará análise espacial em dados nesse banco de dados. Este banco de dados conterá tabelas com informações fictícias sobre clientes e zonas de alagamento. A partir destas informações você pode efetuar testes com o Spatial Extender e determinar quais clientes estão correndo risco de sofrer danos decorrentes de um alagamento.

Com o programa de amostra, você pode:

- Consultar as etapas geralmente requeridas para criar e manter um banco de dados ativado espacialmente.
- Entender como chamar procedimentos armazenados espaciais a partir de um programa aplicativo.
- Recortar e colar código de amostra em seus próprios aplicativos.

Utilize o seguinte programa de amostra para codificar tarefas para o DB2 Spatial Extender. Por exemplo, suponha que você crie um aplicativo que utiliza a interface do banco de dados para chamar procedimentos armazenados do DB2 Spatial Extender. A partir do programa de amostra, você pode copiar código para personalizar seu aplicativo. Se não estiver familiarizado com as etapas de programação para o DB2 Spatial Extender, você poderá executar o programa de amostra, que mostra cada etapa detalhadamente. Para obter instruções sobre como executar o programa de amostra, consulte a seção “Tarefas Relacionadas” no final deste tópico.

A tabela a seguir descreve cada etapa do programa de amostra. Em cada etapa, você executará uma ação e, em muitos casos, reverterá ou irá desfazer essa ação. Por exemplo, na primeira etapa, você ativará o banco de dados espacial e, em seguida, desativará o banco de dados espacial. Desta forma, você se familiarizará com muitos dos procedimentos armazenados do Spatial Extender. Para informações adicionais sobre os procedimentos armazenados responsáveis por cada etapa, consulte “Tarefas relacionadas”, no fim deste tópico.

Tabela 10. Etapas do Programa de Amostra do DB2 Spatial Extender

Etapas	Ação e Descrição
Ativar ou desativar o banco de dados espacial	• Ative o banco de dados espacial Esta é a primeira etapa necessária para utilizar o DB2 Spatial Extender. Um banco de dados que foi ativado para operações espaciais tem um conjunto de tipos espaciais, um conjunto de funções espaciais, um conjunto de predicados espaciais, novos tipos de índices e um conjunto de tabelas de catálogos espaciais e exibições. • Desative o banco de dados espacial Esta etapa geralmente é executada quando você tiver ativado capacidades espaciais para o banco de dados incorreto ou quando não mais precisar executar operações espaciais neste banco de dados. Ao desativar um banco de dados espacial, você remove o conjunto de tipos espaciais, o conjunto de funções espaciais, o conjunto de predicados espaciais, novos tipos de índices e o conjunto de tabelas de catálogos espaciais e exibições associadas a esse banco de dados. • Ative o banco de dados espacial Igual à opção acima.
Criar ou eliminar um sistema de coordenadas	• Crie um sistema de coordenadas denominado NORTH_AMERICAN Esta etapa cria um novo sistema de coordenadas no banco de dados. • Elimine o sistema de coordenadas denominado NORTH_AMERICAN Esta etapa elimina o sistema de coordenadas NORTH_AMERICAN do banco de dados. • Crie um sistema de coordenadas denominado KY_STATE_PLANE Esta etapa cria um novo sistema de coordenadas, KY_STATE_PLANE, que será utilizado pelo sistema de referência espacial criado na etapa seguinte.
Criar ou eliminar um sistema de referência espacial	• Crie um sistema de referência espacial denominado SRSDEMO1 Esta etapa define um novo SRS (Sistema de Referência Espacial) que é utilizado para interpretar as coordenadas. O SRS inclui dados de geometria em um formato que pode ser armazenado em uma coluna de um banco de dados ativado espacialmente. Depois que o SRS estiver registrado para uma coluna espacial específica, as coordenadas que são aplicáveis a ela poderão ser armazenadas na coluna associada da tabela CLIENTES. • Elimine o SRS denominado SRSDEMO1 Esta etapa será executada se você não precisar mais do SRS no banco de dados. Ao eliminar um SRS, você remove a definição do SRS do banco de dados. • Crie o SRS denominado KY_STATE_SRS

Tabela 10. Etapas do Programa de Amostra do DB2 Spatial Extender (continuação)

Etapas	Ação e Descrição
Criar e ocupar as tabelas espaciais	• Crie a tabela CLIENTES • Preencha a tabela CLIENTES A tabela CLIENTES representa dados de negócios que foram armazenados no banco de dados por vários anos. • Altere a tabela CLIENTES incluindo a coluna LOCALIZAÇÃO A instrução ALTER TABLE inclui uma nova coluna (LOCALIZAÇÃO) de tipo ST_Point. Esta coluna será ocupada, efetuando geocoding das colunas de endereço em uma etapa subseqüente. • Crie a tabela ESCRITÓRIOS A tabela ESCRITÓRIOS representa, entre outros dados, a zona de vendas para cada escritório de uma empresa de seguros. Toda a tabela será ocupada pelos dados do atributo a partir de um banco de dados não-DB2 em uma etapa subseqüente. Esta etapa subseqüente envolve a importação de dados de atributo para a tabela ESCRITÓRIOS a partir de um arquivo shape.
Ocupar as colunas	• Codifique geograficamente os dados de endereço da coluna LOCALIZAÇÃO da tabela CLIENTES com o geocoder denominado KY_STATE_GC Esta etapa executa geocoding espacial em batch chamando o utilitário geocoder. O geocoding em batch geralmente é executado quando é necessário efetuar geocode ou efetuar geocode novamente de uma parte significativa da tabela. • Carregue a tabela ESCRITÓRIOS criada anteriormente a partir do arquivo shape, utilizando o sistema de referência espacial KY_STATE_SRS Esta etapa carrega a tabela ESCRITÓRIOS com dados espaciais existentes, na forma de um arquivo shape. Como a tabela ESCRITÓRIOS existe, o utilitário LOAD anexará os novos registros em uma tabela existente. • Crie e carregue a tabela ZONAS DE ALAGAMENTO a partir do arquivo shape, utilizando o sistema de referência espacial KY_STATE_SRS Esta etapa carrega a tabela ZONAS DE ALAGAMENTO com dados existentes, na formato de um arquivo shape. Como a tabela não existe, o utilitário LOAD a criará antes do carregamento dos dados. • Crie e carregue a tabela REGIÕES a partir do arquivo shape, utilizando o sistema de referência espacial KY_STATE_SRS
Registrar ou cancelar registro do geocoder	• Registre o geocoder SAMPLEGC • Cancele o registro do codificador denominado SAMPLEGC • Registre o codificador KY_STATE_GC Estas etapas registram e cancelam o registro do geocoder denominado SAMPLEGC e, em seguida, criam um novo, KY_STATE_GC, para utilização no programa de amostra.

Tabela 10. Etapas do Programa de Amostra do DB2 Spatial Extender (continuação)

Etapas	Ação e Descrição
Criar índices espaciais	• Crie o índice de grade espacial para a coluna LOCALIZAÇÃO da tabela CLIENTES • Elimine o índice de grade espacial para a coluna LOCALIZAÇÃO da tabela CLIENTES • Crie o índice de grade espacial para a coluna LOCALIZAÇÃO da tabela CLIENTES • Crie o índice de grade espacial para a coluna LOCALIZAÇÃO da tabela ESCRITÓRIOS • Crie o índice de grade espacial para a coluna LOCALIZAÇÃO da tabela ZONAS DE ALAGAMENTO • Crie o índice de grade espacial para a coluna LOCALIZAÇÃO da tabela REGIÕES Estas etapas criam o índice de grade espacial para as tabelas CLIENTES, ESCRITÓRIOS, ZONAS DE ALAGAMENTO e REGIÕES.
Ativar geocoding automático	• Configure o geocoder para a coluna LOCALIZAÇÃO da tabela CLIENTES com o codificador KY_STATE_GC Esta etapa associa a coluna LOCALIZAÇÃO da tabela CLIENTES com o geocoder KY_STATE_GC e configura os valores correspondentes para os parâmetros de codificação geográfica. • Ative a codificação geográfica automática para a coluna LOCALIZAÇÃO da tabela CLIENTES Esta etapa ativa a chamada automática do geocoder. A utilização da codificação automática disponibiliza as colunas LOCALIZAÇÃO, RUA, CIDADE, ESTADO e CEP da tabela CLIENTES para serem sincronizadas entre si para subseqüentes operações inserir e atualizar.
Execute operações de inserir, atualizar e excluir na tabela CLIENTES	• Insira alguns registros com uma rua diferente • Atualize alguns registros com um novo endereço • Exclua todos os registros da tabela Estas etapas demonstram operações de inserir, atualizar e excluir nas colunas RUA, CIDADE, ESTADO e CEP da tabela CLIENTES. Após a ativação da codificação geográfica automática, os dados inseridos ou atualizados nessas colunas são codificados automaticamente na coluna LOCALIZAÇÃO. Este processo foi ativado na etapa anterior.
Desativar geocoding automático	• Desative a codificação geográfica automática para a coluna LOCALIZAÇÃO da tabela CLIENTES • Remova a configuração de codificação geográfica para a coluna LOCALIZAÇÃO da tabela CLIENTES • Elimine o índice espacial para a coluna LOCALIZAÇÃO da tabela CLIENTES Estas etapas desativam a chamada automática do geocoder e o índice espacial na preparação para a próxima etapa. A próxima etapa envolve novo geocoding de toda a tabela CLIENTES. **Recomendação:** Se estiver carregando uma grande quantidade de dados geográficos, elimine o índice espacial antes de carregar os dados e recrie-o após o carregamento dos dados.

Tabela 10. Etapas do Programa de Amostra do DB2 Spatial Extender (continuação)

Etapas	Ação e Descrição
Efetuar novo geocode da tabela CLIENTES	• Efetue a codificação geográfica da coluna LOCALIZAÇÃO da tabela CLIENTES novamente com um nível de precisão inferior: 90% em vez de 100% • Recrie o índice espacial para a coluna LOCALIZAÇÃO da tabela CLIENTES • Reative a codificação geográfica automática com um nível de precisão inferior: 90% em vez de 100% Estas etapas executam o geocoder no modo batch, recriam o índice espacial e reativam a codificação geográfica automática com um novo nível de precisão. Esta ação é recomendada quando um administrador espacial observa uma alta taxa de falhas no processo de geocoding. Se o nível de precisão estiver definido como 100%, ele poderá falhar ao efetuar geocode de um endereço porque não conseguirá encontrar um endereço correspondente nos dados de referência. Reduzindo o nível de precisão, o geocoder poderá ser mais bem-sucedido na localização de dados correspondentes. Depois de ser efetuada nova codificação geográfica da tabela em modo batch, a codificação geográfica automática é reativada e o índice espacial é recriado. Isto permite a manutenção incremental do índice espacial e da coluna espacial para subseqüentes operações de inserir e de atualizar.
Criar uma exibição e registrar a coluna espacial na exibição	• Crie uma exibição denominada CLIENTES DE ALTO RISCO, com base na junção das tabelas CLIENTES e ZONAS DE ALAGAMENTO • Registre a coluna espacial da exibição Estas etapas criam uma exibição e registram sua coluna espacial.
Executar análise espacial	• Localize o número de clientes atendidos por cada região (ST_Within) • Para escritórios e clientes com a mesma região, localize o número de clientes que estão em uma distância específica de cada escritório (ST_Within, ST_Distance) • Para cada região, localize a renda média e adicional de cada cliente (ST_Within) • Localize o número de zonas de alagamento encontradas em cada zona de escritórios (ST_Overlaps) • Localize o escritório mais próximo de uma localização de cliente específica, assumindo que ele esteja localizado no centróide da zona de escritórios (ST_Distance) • Localize os clientes cuja localização esteja próxima do limite de uma zona de alagamento específica (ST_Buffer, ST_Intersects) • Localize os clientes de alto risco em um escritório especificado (ST_Within) Todas estas etapas utilizam o procedimento armazenado sqlRunSpatialQueries. Estas etapas executam análise espacial utilizando os predicados espaciais e funções no DB2 SQL. O otimizador de consultas do DB2 explora o índice espacial nas colunas espaciais para aprimorar o desempenho de consultas sempre que possível.

Tabela 10. Etapas do Programa de Amostra do DB2 Spatial Extender (continuação)

Etapas	Ação e Descrição
Exportar dados espaciais para arquivos shape	• Exporte a exibição CLIENTES DE ALTO RISCO para os arquivos shape Esta etapa mostra um exemplo de exportação da exibição CLIENTES DE ALTO RISCO para arquivos shape. Exportar dados de um formato de banco de dados para outro formato de arquivo, permite que as informações sejam utilizadas por outras ferramentas (como o ArcExplorer para DB2). Esta etapa está incluída no programa runGseDemo.c, mas foi transformada em comentário apenas para referência. Você pode modificar o programa de amostra para especificar a localização do arquivo shape de exportação e executar novamente o programa de amostra.
Exportar e importar arquivos SDE	• Exporte a tabela CLIENTES para um arquivo de transferência SDE • Importe os dados do recém-exportado arquivo de transferência SDE Estas etapas mostram exemplos de exportação e importação de arquivos de transferência SDE. Estas etapas estão incluídas no programa runGseDemo.c, mas foram transformadas em comentários apenas para referência. Você pode modificar o programa de amostra para especificar a localização do arquivo SDE de exportação e executar novamente o programa de amostra.

Tarefas Relacionadas:

- "Verificando a instalação do Spatial Extender" na página 39
- "Resolução de Problemas de Instalação" na página 40
- "Gerando Aplicativos para o DB2 Spatial Extender" na página 135
- "Chamando Procedimentos Armazenados do DB2 Spatial Extender a Partir de um Aplicativo" na página 136
- "Incluindo o arquivo de cabeçalho do DB2 Spatial Extender em aplicativos espaciais" na página 135

Capítulo 15. Identificando Problemas do DB2 Spatial Extender

Se você encontrar um problema ao trabalhar com o DB2 Spatial Extender, será necessário determinar a causa do problema. Você pode resolver os problemas com o DB2 Spatial Extender destas maneiras:

- Você pode utilizar informações de mensagens para diagnosticar o problema.
- Ao trabalhar com procedimentos armazenados e funções do Spatial Extender, o DB2 retorna informações sobre o êxito ou falha do procedimento armazenado ou função. As informações retornadas serão um código de mensagem (em forma de um inteiro), texto de mensagem, ou ambos, dependendo da interface utilizada para trabalhar com o DB2 Spatial Extender.
- Você pode exibir o arquivo de notificação de administração do DB2, que registra informações de diagnóstico sobre erros.
- Se você encontrar um problema do Spatial Extender recorrente e que pode ser reproduzido, o representante de suporte ao cliente IBM pedirá que você utilize o utilitário de rastreio do DB2 para ajudá-lo a diagnosticar o problema.

Este capítulo discute cada uma destas abordagens.

Como Interpretar Mensagens do DB2 Spatial Extender

Você pode trabalhar com o DB2® Spatial Extender utilizando quatro interfaces diferentes:

- Procedimentos armazenados do DB2 Spatial Extender
- Funções do DB2 Spatial Extender
- CLP (Processador da Linha de Comandos) do DB2 Spatial Extender
- Centro de Controle do DB2

Todas as interfaces retornam mensagens do DB2 Spatial Extender para ajudá-lo a determinar se a operação espacial solicitada foi concluída com êxito ou resultou em um erro.

A tabela a seguir explica cada parte deste texto da mensagem do DB2 Spatial Extender de amostra:

```
GSE0000I: A operação foi concluída com êxito.
```

Tabela 11. As Partes do Texto da Mensagem do DB2 Spatial Extender

Parte do Texto da Mensagem	Descrição
GSE	O identificador da mensagem. Todas as mensagens do DB2 Spatial Extender começam com o prefixo de três letras GSE.
0000	O número da mensagem. Um número de quatro dígitos que varia de 0000 a 9999.

Tabela 11. As Partes do Texto da Mensagem do DB2 Spatial Extender (continuação)

Parte do Texto da Mensagem	Descrição
I	O tipo de mensagem. Uma única letra que indica a gravidade da mensagem: **C** Mensagens de erro críticas **N** Mensagens de erro não críticas **W** Mensagens de aviso **I** Mensagens informativas
A operação foi concluída com êxito.	A explicação da mensagem.

A explicação que aparece no texto da mensagem é uma explicação breve. Você pode recuperar informações adicionais sobre a mensagem que incluem explicação detalhada e sugestões para evitar ou corrigir o problema. Para exibir estas informações adicionais:

1. Abra um prompt de comandos do sistema operacional.
2. Digite o comando de ajuda do DB2 com o identificador da mensagem e o número da mensagem para exibir informações adicionais sobre a mensagem. Por exemplo:

   ```
   DB2 "? GSEnnnn"
   ```

   em que *nnnn* é o número da mensagem.

Você pode digitar o identificador da mensagem GSE e a letra indicando o tipo de mensagem em maiúsculas ou minúsculas. Digitar `DB2 "? GSE0000I"` produzirá o mesmo resultado que digitar `db2 "? gse0000i"`.

Você pode omitir a letra após o número da mensagem quando digitar o comando. Por exemplo, digitar `DB2 "? GSE0000"` produzirá o mesmo resultado que digitar `DB2 "? GSE0000I"`.

Suponha que o código da mensagem seja `GSE4107N`. Quando digitar `DB2 "? GSE4107N"` no prompt de comandos, serão exibidas as seguintes informações:

```
GSE4107N O valor de tamanho da grade "<tamanho_da_grade>"
não é válido no local em que é utilizado.

Explicação: O tamanho da grade especificado "<tamanho_da_grade>" não é válido.

 Foi feita uma das seguintes especificações inválidas quando o índice de grade
 foi criado com a instrução CREATE INDEX:

  - Um número menor do que 0 (zero) foi especificado como o tamanho da
    grade para o primeiro, segundo ou terceiro nível de grade.

  - 0 (zero) foi especificado como o tamanho da grade para o primeiro
    nível de grade.

  - O tamanho da grade especificado para o segundo nível de grade é menor
    do que o tamanho da grade do primeiro nível de grade, mas não é 0 (zero).

  - O tamanho da grade especificado para o terceiro nível de grade é
    menor do que o tamanho da grade do segundo nível de grade, mas não
    é 0 (zero).
```

```
  - O tamanho da grade especificado para o terceiro nível de grade é maior
    do que 0 (zero) mas o tamanho da grade especificado para o segundo nível
    de grade é 0 (zero).

Resposta do Usuário: Especifique um valor válido para o tamanho da grade.

msgcode: -4107

sqlstate: 38SC7
```

Se as informações forem muito extensas para serem exibidas em uma única tela e seu sistema operacional suportar o programa executável **more** e canais, digite este comando:

```
db2 "? GSEnnnn" | more
```

A utilização do programa **more** forçará uma pausa na exibição após cada tela de dados para que você possa ler as informações.

Conceitos Relacionados:

- "Parâmetros de Saída de Procedimentos Armazenados do DB2 Spatial Extender" na página 147
- "Mensagens de Funções do DB2 Spatial Extender" na página 149
- "Mensagens de CLP do DB2 Spatial Extender" na página 151
- "Mensagens do Centro de Controle do DB2" na página 153
- "O Arquivo de Notificação de Administração" na página 155

Tarefas Relacionadas:

- "Rastreando Problemas no DB2 Spatial Extender com o Comando db2trc" na página 154

Referência Relacionada:

- "GSE messages" na publicação *Message Reference Volume 1*

Parâmetros de Saída de Procedimentos Armazenados do DB2 Spatial Extender

Os procedimentos armazenados do DB2® Spatial Extender são chamados *implicitamente* quando você ativa e utiliza o Spatial Extender a partir do Centro de Controle do DB2 ou quando utiliza o CLP do DB2 Spatial Extender (db2se). Você pode chamar procedimentos armazenados *explicitamente* em um programa aplicativo ou a partir da linha de comandos do DB2.

Este tópico descreve como diagnosticar problemas quando procedimentos armazenados são chamados explicitamente em programas aplicativos ou a partir da linha de comandos DB2. Para diagnosticar procedimentos armazenados chamados implicitamente, utilize as mensagens retornadas pelo CLP do DB2 Spatial Extender ou as mensagens retornadas pelo Centro de Controle do DB2. Estas mensagens são discutidas em tópicos separados.

Os procedimentos armazenados do DB2 Spatial Extender têm dois parâmetros de saída: o código da mensagem (msg_code) e o texto da mensagem (msg_text). Os valores de parâmetros indicam o êxito ou falha de um procedimento armazenado.

msg_code

O parâmetro msg_code é um inteiro, que pode ser positivo, negativo ou

zero (0). Os números positivos são utilizados para avisos, os números negativos são utilizados para erros (críticos e não críticos) e zero (0) é utilizado para mensagens informativas.

O valor absoluto de msg_code está incluído em msg_text como o número da mensagem. Por exemplo

- Se msg_code for 0, o número da mensagem será 0000.
- Se msg_code for –219 , o número da mensagem será 0219. O msg_code negativo indica que uma mensagem é um erro crítico ou não crítico.
- Se msg_code for +1036, o número da mensagem será 1036. O número de msg_code positivo indica que a mensagem é um aviso.

Os números de msg_code para procedimentos armazenados do Spatial Extender estão divididos nas três categorias mostradas na tabela a seguir:

Tabela 12. Códigos de Mensagens de Procedimentos Armazenados

Códigos	Categoria
0000 – 0999	Mensagens comuns
1000 – 1999	Mensagens administrativas
2000 – 2999	Mensagens de importação e exportação

msg_text

O parâmetro msg_text é composto pelo identificador da mensagem, pelo número da mensagem, pelo tipo de mensagem e pela explicação. Um exemplo de um valor de msg_text de procedimento armazenado é:

```
GSE0219N   Uma instrução EXECUTE IMMEDIATE
           falhou. SQLERROR = "<sql-error>".
```

A explicação que aparece no parâmetro msg_text é a explicação breve. Você pode recuperar informações adicionais sobre a mensagem que incluem explicação detalhada e sugestões para evitar ou corrigir o problema.

Para obter uma explicação detalhada das partes do parâmetro msg_text, e informações sobre como recuperar informações adicionais sobre a mensagem, consulte o tópico: Como interpretar mensagens do DB2 Spatial Extender.

Trabalhando com procedimentos armazenados em aplicativos:

Quando chamar um procedimento armazenado do DB2 Spatial Extender a partir de um aplicativo, você receberá msg_code e msg_text como parâmetros de saída. Você pode:

- Programar seu aplicativo para retornar os valores de parâmetros de saída ao usuário do aplicativo.
- Executar alguma ação com base no tipo de valor de msg_code retornado.

Trabalhando com procedimentos armazenados a partir da linha de comandos do DB2:

Quando chamar um procedimento armazenado do DB2 Spatial Extender a partir da linha de comandos do DB2, você receberá os parâmetros de saída msg_code e msg_text. Estes parâmetros de saída indicam o êxito ou falha do procedimento armazenado.

Suponha que você se conecte a um banco de dados e deseja chamar o procedimento armazenado ST_disable_db. O exemplo abaixo utiliza um comando DB2 CALL para desativar o banco de dados para operações espaciais e mostra os resultados dos valores de saída. É utilizado um valor de parâmetro force 0, junto com dois pontos de interrogação no final do comando CALL para representar os parâmetros de saída msg_code e msg_text. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

```
call db2gse.st_disable_db(0, ?, ?)

  Valor Parâmetros de Saída
  -------------------------
  Nome Parâmetro  : MSGCODE
  Valor Parâmetro : 0

  Nome Parâmetro  : MSGTEXT
  Valor Parâmetro : GSE0000I  A operação foi concluída com êxito.

  Status Retorno = 0
```

Suponha que o msg_text retornado seja GSE2110N. Utilize o comando de ajuda do DB2 para exibir mais informações sobre a mensagem. Por exemplo:

```
"? GSE2110"
```

São exibidas as seguintes informações:

```
GSE2110N    O sistema de referência espacial para
            a geometria na linha "<número_da_linha>" é inválido.
            O identificador numérico do sistema de referência espacial
            é "<id-srs>".

Explicação: Na linha número_da_linha, a geometria a
ser exportada utiliza um sistema de referência espacial inválido.
A geometria não pode ser exportada.

Resposta ao Usuário: Corrija a geometria indicada ou
exclua a linha da operação de exportação
modificando a instrução SELECT de acordo.

msg_code: -2110

sqlstate: 38S9A
```

Conceitos Relacionados:

- “Como Interpretar Mensagens do DB2 Spatial Extender” na página 145
- “Mensagens de Funções do DB2 Spatial Extender” na página 149
- “Mensagens de CLP do DB2 Spatial Extender” na página 151
- “Mensagens do Centro de Controle do DB2” na página 153

Referência Relacionada:

- “GSE messages” na publicação *Message Reference Volume 1*

Mensagens de Funções do DB2 Spatial Extender

As mensagens retornadas pelas funções do DB2® Spatial Extender geralmente são incorporadas em uma mensagem SQL. O SQLCODE retornado na mensagem indica se ocorreu um erro com a função ou se um aviso está associado à função. Por exemplo:

- O SQLCODE –443 (número de mensagem SQL0443) indica que ocorreu um erro na função.
- O SQLCODE +462 (número de mensagem SQL0462) indica que um aviso está associado à função.

A tabela a seguir explica as partes importantes desta mensagem de exemplo:

```
DB21034E  O comando foi processado como uma instrução SQL porque
não era um comando válido do Processador da Linha de Comandos.
Durante o processamento de SQL,
ele retornou: SQL0443N  A rotina "DB2GSE.GSEGEOMFROMWKT"
(nome específico "GSEGEOMWKT1") retornou um erro
SQLSTATE com o texto de diagnóstico "GSE3421N  O polígono não está fechado.".
SQLSTATE=38SSL
```

Tabela 13. As Partes Importantes de Mensagens de Funções do DB2 Spatial Extender

Parte da Mensagem	Descrição
SQL0443N	O SQLCODE indica o tipo de problema.
GSE3421N	O número da mensagem e o tipo de mensagem do DB2 Spatial Extender. Os números de mensagens para funções vão de GSE3000 a GSE3999. Além disso, as mensagens comuns podem ser retornadas quando você trabalha com funções do DB2 Spatial Extender. Os números de mensagens para mensagens comuns vão de GSE0001 a GSE0999.
O polígono não está fechado	A explicação da mensagem do DB2 Spatial Extender.
SQLSTATE=38SSL	Um código SQLSTATE que identifica ainda mais o erro. É retornado um código SQLSTATE para cada instrução ou linha. • Os códigos SQLSTATE para erros de funções do Spatial Extender são 38S*xx*, em que cada *x* é uma letra ou um número. • Os códigos SQLSTATE para avisos de funções do Spatial Extender são 01HS*x*, em que o *x* é uma letra ou um número.

Um exemplo de uma mensagem de erro SQL0443:

Suponha que você tenha tentado inserir os valores para um polígono na tabela POLYGON_TABLE, conforme mostrado a seguir:

```
INSERT INTO polygon_table ( geometry )
VALUES ( ST_Polygon ( 'polygon (( 0 0, 0 2, 2 2, 1 2)) ') )
```

Isto resulta em uma mensagem de erro porque você não forneceu o valor final para fechar o polígono. A mensagem de erro retornada é:

```
DB21034E  O comando foi processado como uma instrução SQL porque
não era um comando válido do Processador da Linha de Comandos.
Durante o processamento de SQL,
ele retornou: SQL0443N  A rotina "DB2GSE.GSEGEOMFROMWKT"
(nome específico "GSEGEOMWKT1") retornou um erro
SQLSTATE com o texto de diagnóstico "GSE3421N  O polígono não está fechado.".
SQLSTATE=38SSL
```

O número da mensagem SQL `SQL0443N` indica que ocorreu um erro e a mensagem inclui o texto da mensagem do Spatial Extender `GSE3421N O polígono não está fechado`.

Quando receber este tipo de mensagem:

1. Localize o número da mensagem GSE na mensagem de erro do DB2 ou SQL.
2. Utilize o comando help do DB2 (DB2 ?) para ver a explicação da mensagem do Spatial Extender e a resposta ao usuário. Utilizando o exemplo acima, digite o seguinte comando em um prompt da linha de comandos do sistema operacional:

   ```
   DB2 "? GSE3421"
   ```

A mensagem será repetida, junto com uma explicação detalhada e a resposta ao usuário recomendada.

Conceitos Relacionados:

- "Como Interpretar Mensagens do DB2 Spatial Extender" na página 145
- "Parâmetros de Saída de Procedimentos Armazenados do DB2 Spatial Extender" na página 147
- "Mensagens de CLP do DB2 Spatial Extender" na página 151
- "Mensagens do Centro de Controle do DB2" na página 153

Referência Relacionada:

- "GSE messages" na publicação *Message Reference Volume 1*

Mensagens de CLP do DB2 Spatial Extender

O CLP do DB2® Spatial Extender (db2se) retorna mensagens para:

- Procedimentos armazenados, se chamados implicitamente.
- Informações de formatos, se você chamou o programa de subcomando **shape_info** a partir do CLP do DB2 Spatial Extender. Estas são mensagens informativas.
- Operações de migração.
- Operações de importação e exportação de formatos para e a partir do cliente.

Exemplos de Mensagens de Procedimento Armazenado Retornadas pelo CLP do DB2 Spatial Extender:

A maioria das mensagens retornadas pelo CLP do DB2 Spatial Extender destinam-se aos procedimentos armazenados do DB2 Spatial Extender. Quando chamar um procedimento armazenado a partir do CLP do DB2 Spatial Extender, você receberá um texto de mensagem que indica o êxito ou falha do procedimento armazenado.

O texto da mensagem é composto pelo identificador da mensagem, pelo número da mensagem, pelo tipo da mensagem e pela explicação. Por exemplo, se você ativar um banco de dados utilizando o comando `db2se enable_db testdb`, o texto da mensagem retornada pelo CLP do Spatial Extender será:

```
Ativando o banco de dados. Aguarde...

GSE1036W   A operação foi concluída com êxito. Mas alguns parâmetros de
           configuração do banco de dados e do gerenciador do parâmetros
           de configuração do banco de dados devem ser aumentados.
```

Da mesma forma, se você desativar um banco de dados utilizando o comando `db2se disable_db testdb`, o texto da mensagem retornada pelo CLP do Spatial Extender será:

```
GSE0000I  A operação foi concluída com êxito.
```

A explicação que aparece no texto da mensagem é uma explicação breve. Você pode recuperar informações adicionais sobre a mensagem que incluem explicação detalhada e sugestões para evitar ou corrigir o problema. As etapas para recuperar estas informações e uma explicação detalhada de como interpretar as partes do texto da mensagem são discutidas em um tópico separado.

Se estiver chamando procedimentos armazenados por meio de um programa aplicativo ou da linha de comandos do DB2, há um tópico separado que discute como diagnosticar os parâmetros de saída.

Exemplo de Mensagens de Informações de Formatos Retornadas pelo CLP do Spatial Extender:

Suponha que você tenha decidido exibir informações para um arquivo modelo denominado `office`. Utilizando o CLP do Spatial Extender (db2se), você pode emitir este comando:

```
db2se shape_info -fileName /tmp/offices
```

Este é um exemplo das informações que são exibidas:

```
Informações do arquivo modelo
------------------------------
Código do arquivo                           = 9994
Comprimento do arquivo (palavras de 16 bits) = 484
Versão do arquivo modelo                    = 1000
Tipo de formato                             = 1 (ST_POINT)
Número de registros                         = 31

Coordenada X mínima = -87.053834
Coordenada X máxima = -83.408752
Coordenada Y mínima = 36.939628
Coordenada Y máxima = 39.016477
Formatos não têm coordenadas Z.
Formatos não têm coordenadas M.

O arquivo de índice de formas (extensão .shx) está presente.

Informações do arquivo de atributos
-----------------------------------
Código do arquivo dBase          = 3
Data da última atualização       = 1901-08-15
Número de registros              = 31
Número de bytes no cabeçalho     = 129
Número de bytes em cada registro = 39
Número de colunas                = 3

Nro da Coluna  Nome Coluna      Tipo Dados      Compr.  Decimal

           1   NOME             C ( Caractere)      16        0
           2   FUNCIONÁ.        N (  Numérico™)      11        0
           3   ID               N (  Numérico)      11        0

Definição do sistema de coordenadas: "GEOGCS["GCS_North_American_1983",
DATUM["D_North_American_1983",SPHEROID["GRS_1980",6378137,298.257222101]],
PRIMEM["Greenwich",0],UNIT["Degree",0.01745329251994329 5]]"
```

Exemplos de Mensagens de Migração Retornadas pelo CLP do Spatial Extender:

Quando chamar comandos que executam operações de migração, são retornadas mensagens que indicam o êxito ou a falha dessa operação.

Suponha que você tenha chamado a migração do banco de dados mydb utilizando o comando db2se migrate mydb -messagesFile /tmp/migrate.msg. O texto da mensagem retornada pelo CLP do Spatial Extender será:

```
Migrando banco de dados. Aguarde ...
GSE0000I  A operação foi concluída com êxito.
```

Conceitos Relacionados:

- "Como Interpretar Mensagens do DB2 Spatial Extender" na página 145
- "Parâmetros de Saída de Procedimentos Armazenados do DB2 Spatial Extender" na página 147
- "Mensagens de Funções do DB2 Spatial Extender" na página 149
- "Mensagens do Centro de Controle do DB2" na página 153

Referência Relacionada:

- "GSE messages" na publicação *Message Reference Volume 1*

Mensagens do Centro de Controle do DB2

Quando trabalhar com o DB2® Spatial Extender utilizando o Centro de Controle do DB2, as mensagens aparecerão na janela Mensagem do DB2. A maioria das mensagens que aparecerão são mensagens do DB2 Spatial Extender. Ocasionalmente, você receberá uma mensagem SQL. As mensagens SQL são retornadas quando um erro envolve licença, bloqueio ou quando um serviço DAS não está disponível. As seções a seguir fornecem exemplos de como as mensagens do DB2 Spatial Extender e as mensagens SQL aparecerão no Centro de Controle do DB2.

Mensagens do DB2 Spatial Extender:

Quando você recebe uma mensagem do DB2 Spatial Extender por meio do Centro de Controle, todo o texto da mensagem aparece na área de texto da janela Mensagem do DB2, por exemplo:

```
GSE0219N   Uma instrução EXECUTE IMMEDIATE
           falhou. SQLERROR = "<erro-sql>".
```

Mensagens SQL:

Quando você recebe uma mensagem SQL por meio do Centro de Controle que pertence ao DB2 Spatial Extender:

- O identificador da mensagem, o número da mensagem e o tipo da mensagem aparecem à esquerda da janela Mensagem do DB2, por exemplo: SQL0612N.
- O texto da mensagem aparece na área de texto da janela Mensagem do DB2.

O texto da mensagem que aparece na janela Mensagem do DB2 pode conter o texto da mensagem SQL e o SQLSTATE, ou pode conter o texto da mensagem e a explicação detalhada e resposta do usuário.

Um exemplo de uma mensagem SQL que contém o texto da mensagem SQL e o SQLSTATE é:

```
[IBM][CLI Driver][DB2/NT] SQL0612N "<nome>" é um nome duplicado. SQLSTATE=42711
```

Um exemplo de uma mensagem SQL que contém o texto da mensagem e a explicação detalhada e a resposta do usuário é:

```
SQL8008N
O produto "DB2 Spatial Extender" não tem uma chave de licença
válida instalada e o período de avaliação expirou.

Explicação:
Não foi encontrada uma chave de licença válida e o período de
avaliação expirou.

Resposta ao Usuário:

Instale uma chave de licença para a versão totalmente qualificada
do produto. Você pode obter uma chave de licença para o produto
entrando em contato com o representante IBM® ou representante autorizado.
```

Conceitos Relacionados:

- "Como Interpretar Mensagens do DB2 Spatial Extender" na página 145
- "Parâmetros de Saída de Procedimentos Armazenados do DB2 Spatial Extender" na página 147
- "Mensagens de Funções do DB2 Spatial Extender" na página 149
- "Mensagens de CLP do DB2 Spatial Extender" na página 151

Referência Relacionada:

- "GSE messages" na publicação *Message Reference Volume 1*

Rastreando Problemas no DB2 Spatial Extender com o Comando db2trc

Quando ocorrer um problema recorrente e reprodutível no DB2 Spatial Extender, você pode utilizar o recurso de rastreio do DB2 para capturar informações sobre o problema. O recurso de rastreio do DB2 é ativado pelo comando do sistema **db2trc**. O recurso de rastreio do DB2 pode:

- Rastrear eventos
- Efetuar dump dos dados de rastreio em um arquivo
- Formatar dados de rastreio em um formato legível

Restrições:

Ative este recurso somente quando indicado por um representante de suporte técnico do DB2.

Nos sistemas operacionais UNIX, é necessário possuir autorização SYSADM, SYSCTRL ou SYSMAINT para rastrear uma instância do DB2.

Nos sistemas operacionais Windows, não é necessária autorização especial.

Procedimento:

Para rastrear eventos do DB2 Spatial Extender para a memória, execute estas etapas básicas:

1. Encerre todos os outros aplicativos.
2. Ative o rastreio. O representante de suporte técnico do DB2 fornecerá os parâmetros específicos para esta etapa. O comando básico é:

   ```
   db2trc on
   ```

Restrição: O comando **db2trc** deve ser inserido em um prompt de comandos do sistema operacional ou em um script de shell. Ele não pode ser utilizado na interface da linha de comandos do DB2 Spatial Extender (db2se) ou no DB2 CLP.

Você pode rastrear para a memória ou para um arquivo. O melhor método de rastreio é para a memória. Se o problema que está sendo recriado suspender a estação de trabalho e impedir o dump do rastreio, rastreie para um arquivo.

3. Reproduza o problema.
4. Efetue dump do rastreio para um arquivo. Por exemplo:

   ```
   db2trc dump
   january23trace.dmp
   ```

   Esse comando cria um arquivo (*january23trace.dmp*) no diretório atual, com o nome especificado, e efetua dump das informações de rastreio nesse arquivo.

   Você pode especificar um diretório diferente, incluindo o caminho do arquivo. Por exemplo, para colocar o arquivo de dump no diretório /tmp/spatial/errors, a sintaxe é:

   ```
   db2trc dump /tmp/spatial/errors/january23trace.dmp
   ```

   Efetue dump do rastreio imediatamente após o problema ocorrer.
5. Desative o rastreio. Por exemplo:

   ```
   db2trc off
   ```
6. Formate os dados como um arquivo ASCII. Você pode classificar os dados de duas formas:
   - Utilize a opção **flw** para classificar os dados por processo ou por encadeamento. Por exemplo:

     ```
     db2trc flw
     january23trace.dmp january23trace.flw
     ```
   - Utilize a opção **fmt** para relacionar os eventos em ordem cronológica. Por exemplo:

     ```
     db2trc fmt january23trace.dmp january23trace.fmt
     ```

Conceitos Relacionados:

- "DB2 trace (db2trc)" em *Troubleshooting Guide*
- "Como Interpretar Mensagens do DB2 Spatial Extender" na página 145
- "O Arquivo de Notificação de Administração" na página 155

Referência Relacionada:

- "GSE messages" na publicação *Message Reference Volume 1*

O Arquivo de Notificação de Administração

As informações de diagnóstico sobre os erros são registradas no arquivo de notificação de administração. Essas informações são utilizadas na determinação de problemas e são direcionadas para o suporte técnico do DB2®.

O arquivo de notificação de administração contém informações em texto registradas pelo DB2 e pelo DB2 Spatial Extender. Ele está localizado no diretório especificado pelo parâmetro de configuração do gerenciador do banco de dados DIAGPATH. Nos sistemas Windows® NT, Windows 2000 e Windows XP, o arquivo de notificação de administração do DB2 está no log de eventos e pode ser revisto utilizando o Visualizador de Eventos do Windows.

As informações que o DB2 registra no log de administração são determinadas pelas definições DIAGLEVEL e NOTIFYLEVEL.

Utilize um editor de texto para exibir o arquivo na máquina na qual você suspeita que ocorreu um problema. Os eventos mais recentes registrados são os últimos do arquivo. Geralmente, cada entrada contém as seguintes partes:

- Uma data e hora.
- A localização que relata o erro. Os identificadores do aplicativo permitem corresponder entradas relativas a um aplicativo nos logs de servidores e clientes.
- Uma mensagem de diagnóstico (geralmente iniciando com ″DIA″ ou ″ADM″) explicando o erro.
- Dados de suporte disponíveis, como estruturas de dados SQLCA e ponteiros para a localização de arquivos dump ou trap extra.

Se o banco de dados estiver com comportamento normal, esse tipo de informação não é importante e pode ser ignorado.

O arquivo de notificação de administração cresce continuamente. Quando ficar muito grande, faça backup e apague-o. Um novo arquivo será gerado automaticamente na próxima vez que o sistema precisar dele.

Conceitos Relacionados:

- “Interpreting the administration logs” em *Troubleshooting Guide*
- “Como Interpretar Mensagens do DB2 Spatial Extender” na página 145

Tarefas Relacionadas:

- “Rastreando Problemas no DB2 Spatial Extender com o Comando db2trc” na página 154

Referência Relacionada:

- “GSE messages” na publicação *Message Reference Volume 1*

Parte 4. Utilizando o DB2 Geodetic Extender

Capítulo 16. DB2 Geodetic Extender

Este capítulo apresenta o DB2 Geodetic Extender explicando sua finalidade, descrevendo quando utilizá-lo e explicando conceitos geodésicos.

DB2 Geodetic Extender

O DB2® Geodetic Extender permite tratar a Terra como um globo. Utilizando os mesmos tipos de dados e funções espaciais como para qualquer outra operação do Spatial Extender, é possível utilizar o Geodetic Extender para executar consultas ininterruptas de dados em torno dos pólos e de dados que cruzam o meridiano de 180 graus. É possível manter dados que são referidos para uma localização precisa na superfície da Terra.

O Geodetic Extender é denominado para a disciplina conhecida como *geodésia*. Geodésia é o estudo do tamanho e forma da Terra (ou de qualquer corpo modelado por um elipsóide, como o Sol ou uma esfera celeste). O Geodetic Extender foi projetado para tratar objetos definidos na superfície da Terra com um alto grau de precisão.

Para obter esta precisão, o Geodetic Extender utiliza um sistema de coordenadas de latitude e longitude em um modelo elipsoidal da Terra ou em um *datum geodésico*, em vez de um sistema de coordenadas planar, *x*- e *y*. Um modelo elipsoidal evita distorções, inexatidões e imprecisões que podem ser introduzidas utilizando projeções planas. Para obter informações adicionais, consulte "Latitude e Longitude Geodésicas" na página 161, "Sistema de Coordenadas Geográficas" na página 59 e "Sistema de Coordenadas Projetadas" na página 64.

Para acessar operações geodésicas em vez de espaciais, é necessário definir um sistema de referência espacial geodésico para seus dados. Estes sistemas possuem SRIDs (spatial reference system IDs, IDs do sistema de referência espacial) no intervalo de 2000000000 a 2000001000. O Geodetic Extender fornece 318 sistemas de referência espacial geodésicos predefinidos.

O DB2 Spatial Extender deve ser instalado antes da utilização do DB2 Geodetic Extender. O DB2 Geodetic Extender pode ser comprado separadamente do DB2 Spatial Extender e é necessário comprar uma licença separada para o Geodetic Extender.

Conceitos Relacionados:

- "Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender" na página 160
- "Datums Geodésicos" na página 160

Tarefas Relacionadas:

- "Configurando e Ativando o DB2 Geodetic Extender" na página 167

Referência Relacionada:

- "Datums Suportados pelo DB2 Geodetic Extender" na página 213

Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender

O DB2® Spatial Extender e o DB2 Geodetic Extender gerenciam dados do GIS (Geographic Information System) em um banco de dados do DB2. Cada extender utiliza tecnologias principais diferentes que resolvem problemas diferentes e se complementam:

- O Geodetic Extender trata a Terra como um globo. Ele utiliza um sistema de coordenadas de latitude e de longitude em um modelo de Terra elipsoidal. As operações geométricas são precisas, independentemente da localização. Ele é baseado na biblioteca Hipparchus, que é licenciada pela Geodyssey Limited. Consulte http://www.geodyssey.com para obter informações geodésicas adicionais.

  O Geodetic Extender é melhor utilizado para conjuntos de dados globais e aplicativos que cobrem grandes áreas na Terra, em que uma única projeção de mapa não pode fornecer a precisão requerida pelo aplicativo.
- O Spatial Extender trata a Terra como um mapa plano. Ele utiliza a geometria planimétrica (plana), que significa que ele aproxima a superfície redonda da Terra projetando-a em um plano. Esta projeção causa distorções, que podem variar de acordo com a extensão dos dados, mas as distorções geralmente aumentam junto às bordas da região projetada. Cada projeção de mapa plano possui algum tipo de distorção. O Spatial Extender é baseado na biblioteca de formatos ESRI, que é licenciada pela ESRI. Consulte http://www.esri.com para obter informações espaciais adicionais.

  O Spatial Extender é melhor utilizado para conjuntos de dados locais e regionais que são bem representados em coordenadas projetadas e para aplicativos nos quais a precisão da localização não é importante. Por exemplo, uma empresa de plano de saúde talvez queira saber as localizações de hospitais e clínicas em um estado.

Conceitos Relacionados:

- "DB2 Geodetic Extender" na página 159
- "Regiões Geodésicas" na página 164
- "Latitude e Longitude Geodésicas" na página 161
- "Distâncias Geodésicas" na página 162
- "Esferóides Geodésicos" na página 222

Tarefas Relacionadas:

- "Configurando e Ativando o DB2 Geodetic Extender" na página 167

Datums Geodésicos

Um datum geodésico é um sistema de referência que descreve a superfície da Terra. Muitos destes sistemas de referência foram desenvolvidos durante os séculos à medida que a ciência desenvolveu novas ferramentas para medir a Terra. As medidas por terra e por satélite foram utilizadas para criar datums que, por sua vez, são utilizados para criar projeções de mapa plano.

Os datums geodésicos são baseados em uma aproximação do formato geral da Terra por um elipsóide de rotação (também chamado de *esferóide*). Um esferóide é o formato tridimensional descrito por uma elipse quando girado em torno de um

de seus eixos. Para obter informações adicionais sobre esferóides, consulte "Sistema de Coordenadas Geográficas" na página 59.

Cada objeto espacial definido deve referir-se a um datum específico. Você especifica um datum por seu SRID (Spatial Reference System Identifier). Você pode escolher qualquer datum que seja suportado pelo DB2® Geodetic Extender. Estes sistemas possuem SRIDs no intervalo de 2000000000 a 2000001000.

- "Datums Suportados pelo DB2 Geodetic Extender" na página 213 lista os 318 sistemas de referência espacial geodésicos predefinidos fornecidos pelo Geodetic Extender.
- Você também pode definir um novo datum criando um sistema de referência espacial com um ID no intervalo de 2000000318 a 2000001000. Para obter informações adicionais, consulte "Criando um Sistema de Referência Espacial" na página 74.

Restrições: As funções que utilizam mais de um objeto geo-espacial como argumento não podem tratar combinações de datums. O Geodetic Extender não executa conversões de datums.

Conceitos Relacionados:

- "Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender" na página 160
- "Regiões Geodésicas" na página 164
- "Latitude e Longitude Geodésicas" na página 161
- "Distâncias Geodésicas" na página 162
- "Esferóides Geodésicos" na página 222

Tarefas Relacionadas:

- "Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema" na página 68
- "Criando um Sistema de Referência Espacial" na página 74

Referência Relacionada:

- "Datums Suportados pelo DB2 Geodetic Extender" na página 213

Latitude e Longitude Geodésicas

O sistema de referência de coordenadas do DB2 Geodetic Extender utiliza a *latitude* e *longitude geodésicas* para descrever localizações relativas à Terra. A latitude e longitude geodésicas são sempre baseadas em um datum específico.

Latitude Geodésica
A latitude geodésica de um ponto é o ângulo entre o plano equatorial e a linha perpendicular que cruza a linha normal no ponto na superfície da Terra.

Longitude Geodésica
A longitude geodésica é o ângulo no plano equatorial entre a linha *a* que liga o centro da Terra ao meridiano principal e a linha *b* que liga o centro ao meridiano no qual a linha se encontra. Um *meridiano* é um caminho direto na superfície do datum que é a menor distância entre os pólos.

O elipsóide na Figura 17 mostra os ângulos que representam a latitude e longitude geodésicas. O ângulo para a latitude geodésica não é iniciado bem no centro devido ao formato elipsoidal da Terra.

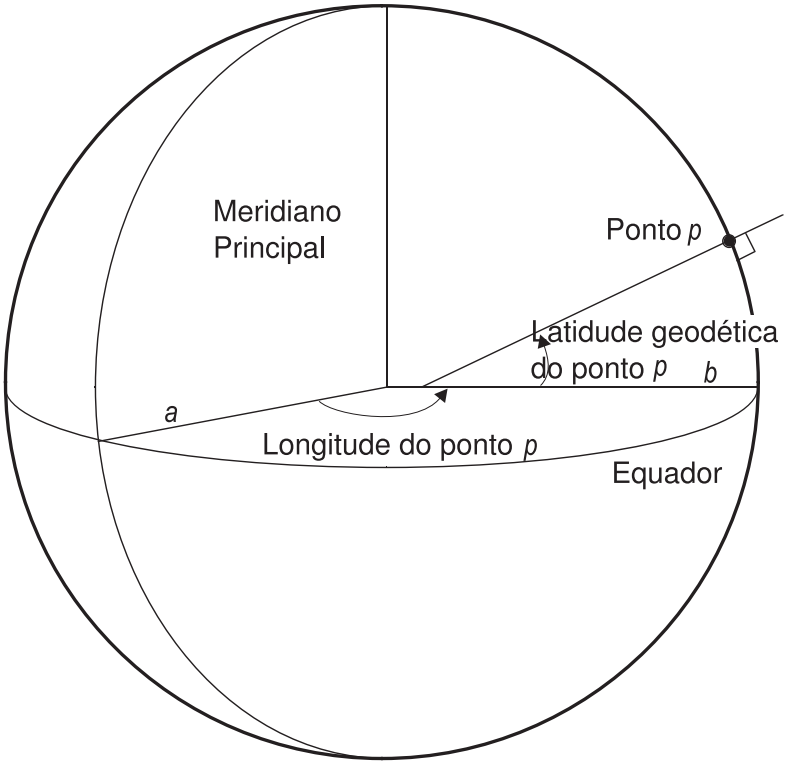


Figura 17. Ângulos de Latitude e Longitude Geodésicas

As coordenadas de latitude e longitude são expressas em graus com uma fração decimal. Existem 360 graus de longitude, começando no meridiano principal (0° de longitude) e prosseguindo em sentido do leste em uma direção positiva de 180° e oeste em valores negativos em –180°. Os graus de latitude começam no equador (0° de latitude) e prosseguem para o Pólo Norte (90° de latitude) e Pólo Sul (–90° de latitude).

Conceitos Relacionados:

- "DB2 Geodetic Extender" na página 159
- "Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender" na página 160
- "Regiões Geodésicas" na página 164
- "Datums Geodésicos" na página 160
- "Distâncias Geodésicas" na página 162
- "Esferóides Geodésicos" na página 222

Distâncias Geodésicas

O DB2® Geodetic Extender calcula a distância entre dois pontos em um *geodésico*. Um geodésico é o caminho mais curto entre dois pontos no formato elipsoidal da Terra e este caminho mais curto pode não seguir uma linha de latitude constante, mesmo que os dois pontos extremos estejam na mesma latitude.

Como os segmentos lineares são calculados como geodésicos, um polígono de quatro pontos com pontos amplamente separados, conforme mostra a Figura 18, pode não incluir a região pretendida. Este polígono cobre uma região com linhas longitudinais que possuem aproximadamente 120 graus separadamente, e os dois pontos superiores possuem os mesmos valores latitudinais e os dois pontos inferiores possuem os mesmos valores latitudinais. O geodésico entre as duas linhas longitudinais segue a curva no formato elipsoidal da Terra. A latitude aumenta no geodésico em 20 graus no meio de qualquer extremidade do geodésico.

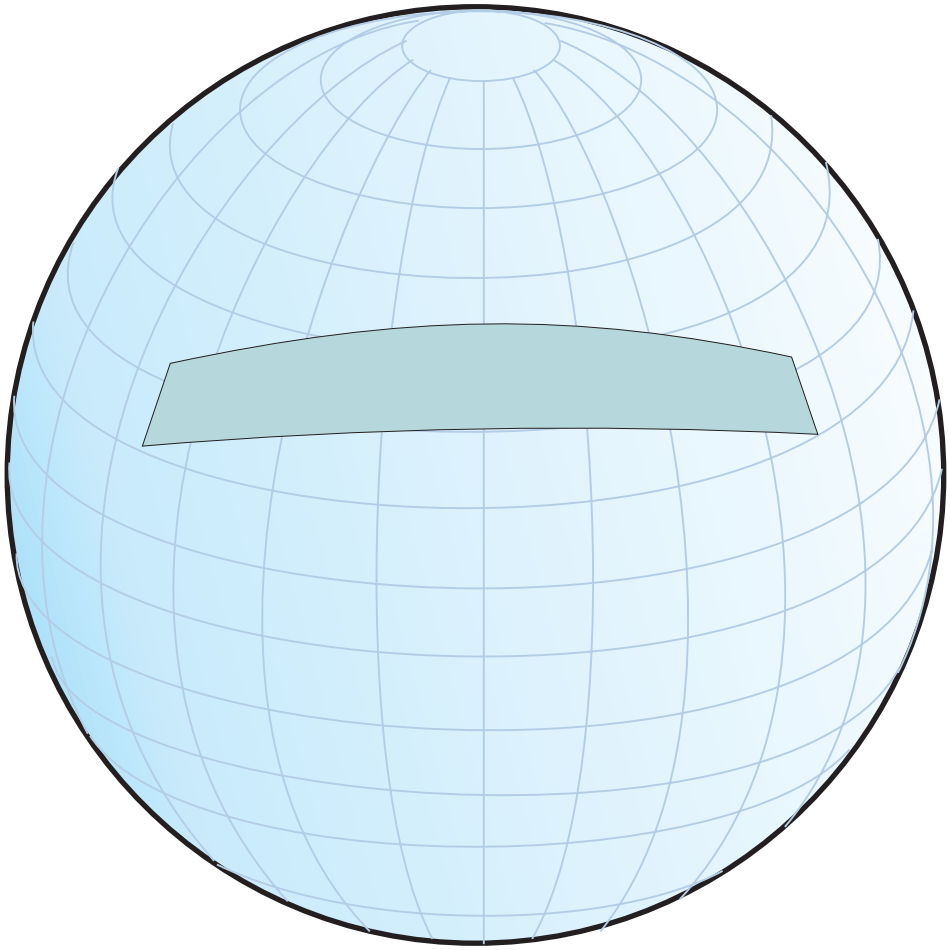


Figura 18. Região Contida em um Polígono com Pontos Amplamente Separados

Para representar um caminho que não seja um geodésico, por exemplo, se desejar que um segmento linear siga uma latitude constante, será necessário inserir pontos intermediários adicionais.

Conceitos Relacionados:

- "Sistema de Coordenadas Geográficas" na página 59

Referência Relacionada:

- "Diferenças em Trabalhar com Representações Planas e Redondas da Terra" na página 199
- "ST_Distance" na página 373

Regiões Geodésicas

Uma região geodésica (polígono) é uma área na superfície da Terra que possui algumas características específicas de um aplicativo. Exemplos de regiões incluem uma área de influência de mercado ou uma área vista por satélite por um período de tempo especificado.

O Geodetic Extender define uma região por uma seqüência ordenada de pontos que formam um anel fechado. A ordem na qual você especifica pontos em um polígono é importante. Conforme você segue um polígono de vértice a vértice na ordem definida, a área à esquerda está dentro do polígono.

Você pode utilizar um tipo de dados ST_Polygon para definir uma região contida em um ou mais anéis, conforme mostra a Figura 19 na página 164. Defina o polígono por coordenadas de latitude e de longitude dos pontos (vértices) que formam seus anéis.

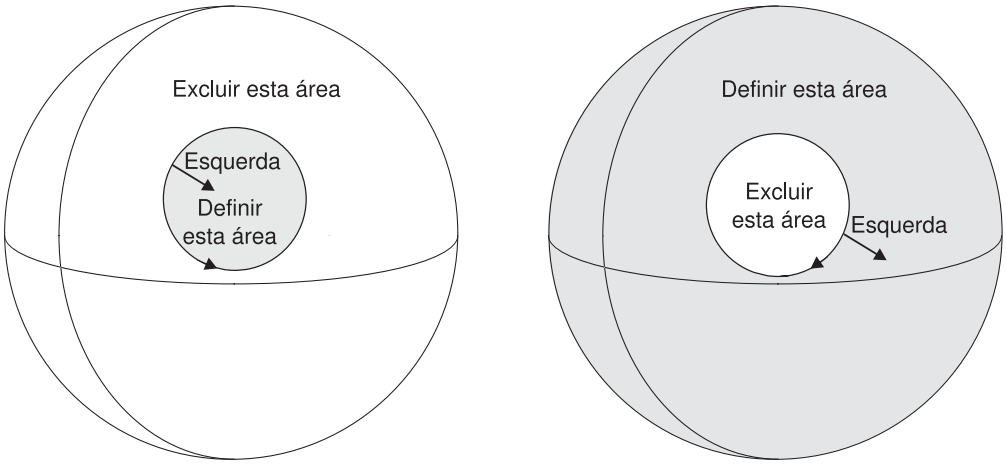


Figura 19. Definindo e Excluindo Áreas

Um anel divide a superfície da Terra em duas regiões: uma região dentro do polígono e uma fora do polígono. O lado esquerdo da Figura 19 mostra um anel com vértices especificados em seqüência anti-horária para que todos os pontos à esquerda fiquem dentro do anel. O lado direito da figura mostra um anel com vértices em seqüência horária para que todos os pontos à esquerda fiquem fora do anel.

Para definir uma região como um polígono, você deve especificar a ordem dos vértices de cada anel para que o interior do polígono fique à sua esquerda quando você passar pelo anel. Para definir uma região excluída, você deve especificar os vértices do anel na ordem oposta, conforme a Figura 20 na página 165 ilustra. O interior do polígono fica sempre à esquerda. A Figura 20 na página 165 mostra dois anéis, um dentro do outro. O anel maior define o limite externo do polígono e é desenhado no sendo anti-horário. O anel menor define o limite interno e é desenhado no sentido horário.

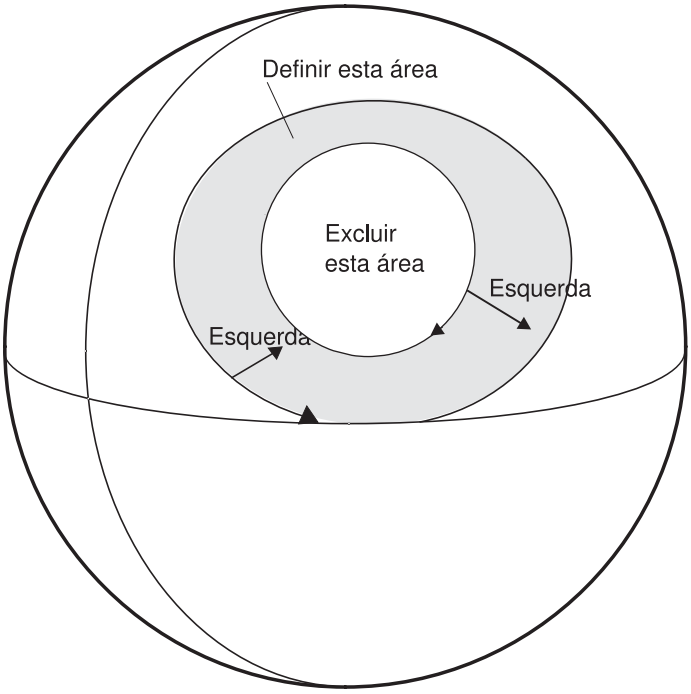


Figura 20. Definindo uma Área com Vários Anéis

Se você criar um polígono que seja maior que um hemisfério, será retornada a seguinte mensagem de aviso. Talvez você realmente queira este polígono maior, mas o aviso é para casos em que você especifica inadvertidamente a ordem incorreta do vértice e resulta em um polígono grande quando você deseja um polígono pequeno.

```
GSE3733W "O polígono cobre mais da metade da terra.
  Verifique a orientação em sentido anto-horário dos
  pontos do vértice."
```

Conceitos Relacionados:

- "Sistema de Coordenadas Projetadas" na página 64
- "Datums Geodésicos" na página 160
- "Sistemas de Referência Espacial" na página 67

Tarefas Relacionadas:

- "Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema" na página 68
- "Criando um Sistema de Referência Espacial" na página 74

Capítulo 17. Configurando o DB2 Geodetic Extender

Este capítulo fornece instruções para configurar o DB2 Geodetic Extender, para migrar do Informix Geodetic DataBlade e para ocupar colunas espaciais com dados geodésicos.

Configurando e Ativando o DB2 Geodetic Extender

O DB2 Geodetic Extender trata a Terra como um globo; enquanto o Spatial Extender trata a superfície redonda da Terra como um mapa plano. Se você instalar o Geodetic Extender, poderá analisar dados espaciais com mais precisão do que em um mapa plano.

Um sistema DB2 Geodetic Extender consiste no DB2 Universal Database, no DB2 Spatial Extender, no DB2 Geodetic Extender e, para a maioria dos aplicativos, um geonavegador.

Recomendação: Para quaisquer informações adicionais ou alteradas para ativar o DB2 Geodetic Extender, consulte *Notas sobre o Release do DB2*.

Pré-requisitos:

Antes de ativar o DB2 Geodetic Extender, você deve:

- Instalar e configurar o DB2 Universal Database™ Enterprise Server Edition Versão 8.2.

  Você deve instalar o DB2 UDB em seu sistema *antes* de instalar o DB2 Spatial Extender e o DB2 Geodetic Extender. Se você pretende utilizar o Centro de Controle do DB2, crie e configure o DAS (DB2 Administration Server). Para obter informações adicionais sobre como criar e configurar o DAS, consulte o *IBM® DB2 Universal Database Administration Guide: Implementation*
- Instalar e configurar o DB2 Spatial Extender.

  O DB2 Geodetic Extender está integrado no mesmo código de biblioteca que o DB2 Spatial Extender. Portanto, o CD de instalação para o Spatial Extender inclui o Geodetic Extender. Os requisitos de espaço em disco para o Spatial Extender incluem o Geodetic Extender. No entanto, você não pode utilizar o Geodetic Extender até que compre e ative uma licença do Geodetic Extender. Para obter informações adicionais, consulte “Requisitos do Sistema para Instalar o Spatial Extender” na página 26 e “Configurando e Instalando o Spatial Extender” na página 25.
- Se você tiver um banco de dados do DB2 Spatial Extender Versão 8.1, será necessário migrá-lo para a Versão 8.2 antes de utilizar o DB2 Geodetic Extender.

  O Geodetic Extender redefine várias funções espaciais e define sistemas de referência espacial geodésicos adicionais para tratar dados geodésicos. O utilitário de migração **migrate_v82** permite que um banco de dados existente ativado espacialmente trate dados geodésicos. Para obter informações adicionais, consulte “Migrando um Banco de Dados Ativado Espacialmente” na página 45.
- Compre uma licença para o DB2 Geodetic Extender.

  Quando comprar uma licença do DB2 Geodetic Extender, você poderá ativar a chave de licença doGeodetic. Entre em contato com o Representante de Vendas se desejar comprar o DB2 Geodetic Extender.

Restrições:

O DB2 Geodetic Extender está licenciado apenas para o DB2 Universal Database™ Enterprise Server Edition Versão 8.2.

Procedimento:

Ative a licença do DB2 Geodetic Extender de uma das seguintes maneiras:

- Utilize o Centro de Licenças no Centro de Controle do DB2. Consulte a ajuda on-line no Centro de Licenças do DB2 para obter informações adicionais sobre como ativar a licença do Geodetic.
- Execute o comando **db2licm**.

Depois de ativar a licença do DB2 Geodetic Extender, "Ocupando Colunas Espaciais com Dados Geodésicos" na página 175.

Conceitos Relacionados:

- "Requisitos do Sistema para Instalar o Spatial Extender" na página 26

Tarefas Relacionadas:

- "Configurando e Instalando o Spatial Extender" na página 25
- "Migrando um Banco de Dados Ativado Espacialmente" na página 45
- "Ocupando Colunas Espaciais com Dados Geodésicos" na página 175

Referência Relacionada:

- "CDs para Dados e Mapas do DB2 Spatial Extender" na página 42

Migrando do Informix Geodetic DataBlade para o DB2 Geodetic Extender

Se você utilizar o IBM Informix Geodetic DataBlade para armazenar e manipular objetos geoespaciais em um banco de dados, poderá migrar seus dados e aplicativos para o IBM DB2 Geodetic Extender com algumas restrições.

Pré-requisitos:

Você deve permitir que os aplicativos do Geodetic DataBlade utilizem tipos de dados e funções do DB2 Geodetic Extender.

Restrições:

Se você estiver utilizando o Informix Geodetic DataBlade, poderá migrar para o DB2 Geodetic Extender se atender os seguintes critérios:

- Utilize apenas os tipos de dados GeoPoint, GeoLineseg, GeoString, GeoRing e GeoPolygon.
- Utilize apenas as funções do Geodetic DataBlade que possuem contrapartes equivalentes ou semi-equivalentes no DB2 Geodetic Extender, conforme descrevem as tabelas abaixo.
- Indexe apenas o componente espacial de GeoObjects; em outras palavras, não indexe intervalos de tempo ou de altitude.

Procedimento:

Para migrar do IBM Informix Geodetic DataBlade para o IBM DB2 Geodetic Extender:

1. Regrave as instruções SQL para utilizar os tipos de dados e funções do DB2 Geodetic Extender. Consulte as tabelas a seguir para obter os tipos de dados e funções correspondentes:
   - Tabela 14
   - Tabela 15 na página 170
   - Tabela 16 na página 170
   - Tabela 17 na página 171
   - Tabela 18 na página 172
   - Tabela 19 na página 172
   - Tabela 20 na página 172
   - Tabela 21 na página 172
   - Tabela 22 na página 173
   - Tabela 23 na página 173
2. Carregue ou importe seus dados para o DB2 Geodetic Extender.
3. Regrave aplicativos que utilizam o Informix ODBC, ESQL/C e JDBC. A Tabela 24 na página 174 mostra a conectividade de cliente correspondente no Geodetic DataBlade e no Geodetic Extender.

Tabela 14. Tipos de Dados Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Tipo de Dados no Informix Geodetic DataBlade	Tipo de Dados Correspondente no DB2 Geodetic Extender	Comentários para Tipos de Dados Semi-equivalentes
GeoBox		Primeiro converta em um GeoPolygon no Geodetic DataBlade, em seguida, utilize ST_Polygon no Geodetic Extender
GeoCircle		Primeiro converta em um GeoPolygon, em seguida, migre para ST_Polygon
GeoEllipse		Primeiro converta em um GeoPolygon, em seguida, migre para ST_Polygon
GeoLineseg	ST_LineString	
GeoObject	ST_Geometry	ST_Geometry e seus subtipos não suportam os tipos de dados GeoAltRange e GeoTimeRange
GeoPoint	ST_Point	
GeoPolygon	ST_MultiPolygon, ST_Polygon	ST_MultiPolygon requer um ponto de fechamento explícito para cada anel. Se um GeoPolygon tiver um anel externo, ele poderá ser mapeado para um ST_Polygon.
GeoRing	ST_LineString	
GeoString	ST_LineString	

Os tipos de dados do Geodetic DataBlade a seguir não possuem um tipo de dados correspondente no Geodetic Extender:

- GeoAltitude
- GeoAltRange
- GeoAngle
- GeoAzimuth

- GeoBox
- GeoCircle
- GeoCoords
- GeoDistance
- GeoEllipse
- GeoLatitude
- GeoLongitude
- GeoTimeRange
- GeoVoronoi

Tabela 15. Funções de Predicados Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Função no Informix Geodetic DataBlade	Função Correspondente no DB2 Geodetic Extender
Contains	ST_Contains
Inside	ST_Within
Intersect	ST_Intersects
Outside	ST_Disjoint
Within	ST_Distance

As funções de predicados do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- Beyond
- Equal
- Nearest

Tabela 16. Funções de Produção Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Função no Informix Geodetic DataBlade	Função Correspondente no DB2 Geodetic Extender	Comentários para Funções Semi-equivalentes
Difference	ST_Difference	ST_Difference suporta pontos além de polígonos
Generalize	ST_Generalize	
Intersection	ST_Intersection	ST_Intersection(line,line) pode resultar em um multiponto. ST_Intersection (line,poly) pode resultar em uma multicadeia. Retorna Vazio para desconectar objetos.
SymDifference	ST_SymDifference	ST_SymDifference suporta pontos além de polígonos
Union	ST_Union	ST_Union suporta pontos e linhas além de polígonos

Tabela 17. Funções do Responsável pelo Acesso no Informix Geodetic DataBlade e no Geodetic Extender

Função no Informix Geodetic DataBlade	Função Correspondente no DB2 Geodetic Extender	Comentários para Funções Semi-equivalentes
Center	ST_MidPoint, ST_PointOnSurface	ST_MidPoint é um semi-substituto para linhas. ST_PointOnSurface é um semi-substituto para polígonos.
Coords	ST_PointN	
Dimension	ST_Dimension	
HasZValue	ST_Is3d	
IsGeoBox	Utilize a expressão IS OF ou ST_GeometryType	
IsGeoCircle	Utilize a expressão IS OF ou ST_GeometryType	
IsGeoEllipse	Utilize a expressão IS OF ou ST_GeometryType	
IsGeoLineseg	Utilize a expressão IS OF ou ST_GeometryType	
IsGeoPoint	Utilize a expressão IS OF ou ST_GeometryType	
IsGeoPolygon	Utilize a expressão IS OF ou ST_GeometryType	
IsGeoRing	Utilize a expressão IS OF ou ST_GeometryType	
IsGeoString	Utilize a expressão IS OF ou ST_GeometryType	
Latitude	ST_Y	
Longitude	ST_X	
NPoints	ST_NumPoints	
NRings	ST_NumGeometries, ST_NumInteriorRing	Utilize ST_NumGeometries para obter o número total de anéis externos e somas ST_NumInteriorRings para cada polígono no conjunto de multipolígonos
Ring	ST_GeometryN, ST_ExteriorRing, ST_InteriorRingN	Utilize ST_GeometryN junto com ST_ExteriorRing e ST_InteriorRingN
SRID	ST_SRID	
Zvalue	ST_Z	

As funções do responsável pelo acesso do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- IsLarge
- IsSmallArea

Tabela 18. Funções do Modificador Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Função no Informix Geodetic DataBlade	Função Correspondente no DB2 Geodetic Extender
SetSRID	ST_SRID

As funções do modificador do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- SetAltRange
- SetAltRangeZ
- SetDist
- SetTimeRange

Tabela 19. Funções de Medida Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Função no Informix Geodetic DataBlade	Função Correspondente no DB2 Geodetic Extender
Area	ST_Area
Distance	ST_Distance
Length	ST_Length, ST_Perimeter

A função de medida VoronoiResolution não possui uma função correspondente no Geodetic Extender.

Tabela 20. Funções de Downcast Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Função no Informix Geodetic DataBlade	Função Correspondente no DB2 Geodetic Extender
GeoBox	Utilize a expressão SQL TREAT
GeoCircle	Utilize a expressão SQL TREAT
GeoEllipse	Utilize a expressão SQL TREAT
GeoLineseg	Utilize a expressão SQL TREAT
GeoPoint	Utilize a expressão SQL TREAT
GeoPolygon	Utilize a expressão SQL TREAT
GeoRing	Utilize a expressão SQL TREAT
GeoString	Utilize a expressão SQL TREAT

Tabela 21. Funções do Construtor Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Função no Informix Geodetic DataBlade	Função Correspondente no DB2 Geodetic Extender
GeoCoords	ST_Point
GeoPoint	ST_Point

As funções do construtor do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- GeoBox

- GeoCircle
- GeoEllipse
- GeoLineseg

Tabela 22. Funções de Diagnóstico Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Função no Informix Geodetic DataBlade	Função Correspondente no DB2 Geodetic Extender
GeoTraceLevel	Recurso de Rastreio do DB2
IsValidGeometry	ST_IsValid

As funções de diagnóstico do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- GeoInRowSize
- GeoOutOfRowSize
- GeoRelease
- GeoTotalSize
- GeoTraceLevelSet
- GeoWarningLevel
- GeoWarningLevelSet
- IsValidSDTS

Tabela 23. Tabelas do Catálogo do Sistema Correspondentes no Informix Geodetic DataBlade e no Geodetic Extender

Tabela do Catálogo do Sistema no Informix Geodetic DataBlade	Exibição do Catálogo Correspondente no DB2 Geodetic Extender
GeoLenUnit	DB2GSE.ST_UNITS_OF_ MEASURE
GeoSpatialRef	DB2GSE.SPATIAL_REF_SYS

As tabelas do catálogo do sistema do Geodetic DataBlade a seguir não possuem uma tabela ou exibição correspondente no Geodetic Extender:

- GeoEllipsoid
- GeoParam
- GeoVoronoi

As funções do parâmetro user–settable do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- GeoParamSessionGet
- GeoParamSessionSet

As funções AltRange do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- AltRange
- Bottom
- Contains
- Equal
- Inside
- Intersect

- IsAny
- Outside
- Top

As funções TimeRange do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- Begin
- Contains
- End
- Equal
- IsAny
- Inside
- Intersect
- Outside
- TimeRange

As funções de elipse do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- Azimuth
- Coords
- Major
- Minor

As funções de círculo do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- Coords
- Radius

As funções aritméticas de ângulo do Geodetic DataBlade a seguir não possuem uma função correspondente no Geodetic Extender:

- Divide
- Minus
- Negate
- Plus
- Times

Tabela 24. Produtos de Conectividade do Cliente Correspondentes no Geodetic DataBlade e no DB2 Geodetic Extender

Conectividade do Cliente no Informix Geodetic DataBlade	Conectividade do Cliente Correspondente no DB2 Geodetic Extender
ESQLC	SQC
ODBC	ODBC
JDBC	JDBC

A conectividade do cliente do Geodetic DataBlade a seguir não possui uma conectividade de cliente correspondente no Geodetic Extender:

- API Java
- LIBMI

Ocupando Colunas Espaciais com Dados Geodésicos

Depois de criar colunas espaciais e registrar as colunas nas quais planeja criar um índice espacial, você estará pronto para ocupar as colunas com dados geodésicos. Você pode fornecer dados geodésicos das seguintes maneiras:

- Importar os seguintes formatos de dados para uma tabela nova ou existente:
  - Shape
  - SDE
- Inserir ou atualizar valores nos seguintes formatos de dados:
  - Shape
  - SDE
  - WKT (Well-Known Text)
  - WKB (Well-Known Binary)
  - GML (Geography Markup Language)

Restrições:

- Para o Spatial Extender Versão 8.2, você não pode utilizar comandos do geocoder ou procedimentos armazenados para converter dados em dados geodésicos.
- Para o comportamento geodésico, utilize sistemas de referência espacial que possuem SRIDs no intervalo de 2.000.000.000 a 2.000.001.000. Para obter informações adicionais, consulte "Sistemas de Referência Espacial" na página 67.
- Os dados shape e os dados de transferência SDE devem estar em um sistema de coordenadas geográficas. Para obter informações adicionais, consulte "Sistema de Coordenadas Geográficas" na página 59.

Procedimento:

O procedimento para importar dados geodésicos é igual para os dados espaciais. Para obter detalhes, consulte "Importando Dados de Formas para uma Tabela Nova ou Existente" na página 88 e "Importando Dados de Transferência SDE para uma Tabela Nova ou Existente" na página 90.

Conceitos Relacionados:

- "Sistemas de Referência Espacial" na página 67
- "Sistema de Coordenadas Geográficas" na página 59

Tarefas Relacionadas:

- "Importando Dados de Formas para uma Tabela Nova ou Existente" na página 88
- "Importando Dados de Transferência SDE para uma Tabela Nova ou Existente" na página 90
- "Registrando Colunas Espaciais" na página 85

Capítulo 18. Índices Geodésicos

Você pode criar índices geodésicos de Voronoi que podem aprimorar o desempenho quando você consultar dados geodésicos. Este capítulo:

- Descreve índices geodésicos de Voronoi
- Descreve estruturas de células de Voronoi e quando você pode selecionar uma estrutura alternativa.
- Explica como criar um índice geodésico de Voronoi.

Índices Geodésicos de Voronoi

O DB2® Geodetic Extender fornece um índice geodésico de Voronoi que acelera o acesso a dados geodésicos. Este índice organiza o acesso a dados geodésicos utilizando disposições de Voronoi da superfície da Terra. Para obter informações adicionais, consulte "Estruturas de Células de Voronoi" na página 178.

O Geodetic Extender calcula o MBC (Minimum Bounding Circle) para cada geometria. O MBC é um círculo que cerca uma geometria geodésica.O índice de Voronoi utiliza estas informações do MBC para organizar dados em uma estrutura de célula. Uma pesquisa utilizando um índice de Voronoi pode passar rapidamente pelos dados organizados para localizar objetos na área de interesse geral e, em seguida, executar testes mais exatos nos próprios objetos. Um índice de Voronoi pode aprimorar o desempenho porque elimina a necessidade de examinar objetos fora da área de interesse. Sem um índice de Voronoi, uma consulta precisaria avaliar cada objeto para localizar os que correspondem aos critérios da consulta.

O otimizador considera um índice geodésico de Voronoi para utilização por todas as consultas que contêm as seguintes funções em sua cláusula WHERE:

- EnvelopesIntersect
- ST_Contains
- ST_Distance
- ST_EnvIntersects
- ST_Intersects
- ST_MBRIntersects
- ST_Within

Para obter informações adicionais, consulte "Funções que Utilizam Índices para Otimizar Consultas" na página 124.

Quando criar um índice geodésico de Voronoi, você pode escolher uma estrutura de célula de Voronoi alternativa. Para obter detalhes, consulte "Considerações para Seleção de uma Estrutura de Célula de Voronoi Alternativa" na página 179.

Conceitos Relacionados:

- "Estruturas de Células de Voronoi" na página 178
- "Considerações para Seleção de uma Estrutura de Célula de Voronoi Alternativa" na página 179
- "Índices de Grade Espaciais" na página 102

Tarefas Relacionadas:

- "Criando Índices Geodésicos de Voronoi" na página 181

Referência Relacionada:

- "Estruturas de Células de Voronoi Fornecidas com o DB2 Geodetic Extender" na página 184
- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Estruturas de Células de Voronoi

Para fazer cálculos de forma eficiente, o DB2® Geodetic Extender subdivide a superfície da Terra em células menores, mais gerenciáveis e semelhantes a favos de mel. Esta subdivisão é conhecida como uma *disposição de Voronoi*, e a estrutura de dados que a descreve é chamada de *Estrutura de célula de Voronoi*. Uma *disposição de Voronoi* é uma estrutura de célula na qual o interior de cada célula consiste em todos os pontos que estão mais próximos de um ponto de entrelaçamento específico do que de qualquer outro ponto de entrelaçamento. As células em uma estrutura de célula de Voronoi são *invólucros convexos*. Um invólucro convexo de um conjunto de pontos é o menor conjunto convexo que inclui os pontos (ou, o menor polígono que define o ″exterior″ de um grupo de pontos). As Estruturas de Células de Voronoi tendem a ser polígonos com formato irregular; o número e a localização de células podem ser ajustados para corresponder à densidade e localização de seus dados espaciais.

Por exemplo, uma estrutura de célula de Voronoi pode subdividir a Terra em polígonos com base na população humana. Onde a população (e os dados) é densa, existem polígonos pequenos. Onde a população é esparsa, existem polígonos grandes.

A Figura 21 na página 179 mostra a estrutura de Voronoi que é baseada na densidade populacional mundial. O Geodetic Extender utiliza esta estrutura de célula em seus cálculos espaciais.

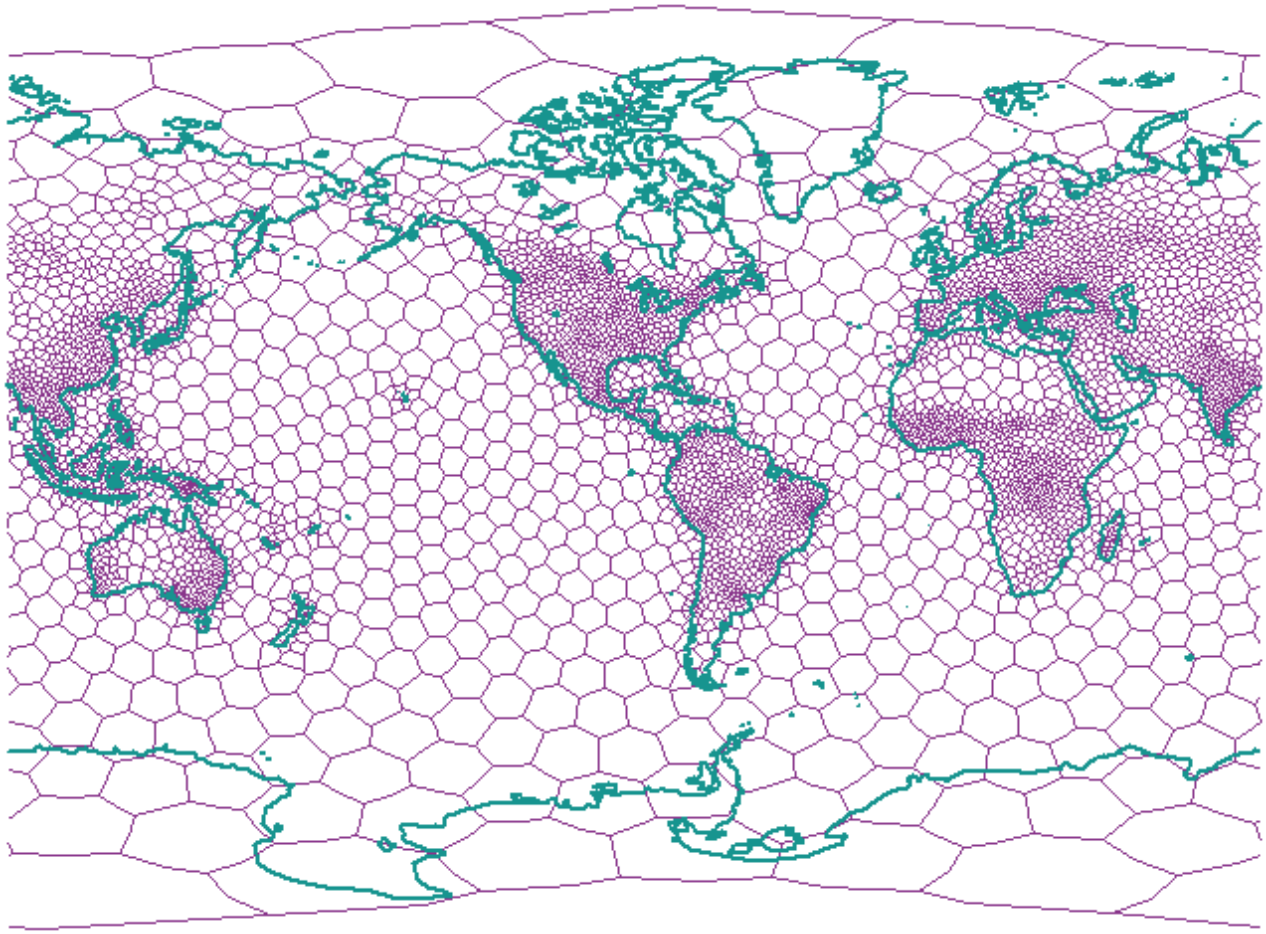


Figura 21. Estrutura de Voronoi Baseada na Densidade Populacional Mundial

Conceitos Relacionados:

- "Índices Geodésicos de Voronoi" na página 177
- "Considerações para Seleção de uma Estrutura de Célula de Voronoi Alternativa" na página 179

Tarefas Relacionadas:

- "Criando Índices Geodésicos de Voronoi" na página 181

Referência Relacionada:

- "Instrução CREATE INDEX para um Índice Geodésico de Voronoi" na página 182
- "Estruturas de Células de Voronoi Fornecidas com o DB2 Geodetic Extender" na página 184

Considerações para Seleção de uma Estrutura de Célula de Voronoi Alternativa

Todas as operações em geometrias geodésicas utilizam um ID de Voronoi 1, que especifica a estrutura de célula de Voronoi baseada na densidade populacional mundial. Quando criar um índice, se seus dados estiverem em cluster em uma ou mais áreas da Terra, como dados de rua para um ou mais países, você poderá

escolher uma estrutura de célula de Voronoi alternativa que possui células menores nas áreas em que seus dados estão localizados (porque a resolução é inversamente proporcional ao tamanho da célula). O DB2® Geodetic Extender fornece várias estruturas de células de Voronoi para indexação que podem ser mais adequadas a seus dados. Para obter uma lista de estruturas alternativas disponíveis e diagramas que ilustram estas estruturas de células, consulte "Estruturas de Células de Voronoi Fornecidas com o DB2 Geodetic Extender" na página 184.

Restrição: Você pode escolher uma estrutura de célula de Voronoi alternativa apenas quando criar um índice geodésico de Voronoi.

A estrutura dodeca04 (ID de Voronoi 12) é mais adequada para dados que são uniformemente distribuídos por toda a superfície da Terra, tais como, imagens de satélite. As células são quase uniformes em tamanho e a resolução do pior caso é de aproximadamente 10 centímetros. É recomendável utilizar uma estrutura de célula de Voronoi diferente da estrutura populacional mundial padrão (ID de Voronoi 1) ou a estrutura dodeca04, se qualquer uma das seguintes condições se aplicar a seus dados ou a seu aplicativo:

Alta Resolução
Se precisar determinar se os objetos com menos de 10 centímetros separados se cruzam, será necessário utilizar uma estrutura de célula de Voronoi que tenha células menores nas regiões em que seus dados estão localizados. A resolução é inversamente proporcional ao tamanho da célula.

Polígonos com Muitos Vértices
Se seus dados consistirem em polígonos que possuem números relativamente grandes de vértices e que são relativamente pequenos em área, é recomendável alternar para uma estrutura de célula de Voronoi que possua mais células em suas regiões de interesse. Se a maioria de seus polígonos tiver 50 vértices ou menos, poderá ser necessário alternar. Se os únicos polígonos em seu conjunto de dados que tiverem muitos vértices tiverem dimensões de continentes, também poderá ser necessário alternar.

Se você tiver muitos polígonos com 3000 vértices que possuem a dimensão de condados dos E.U.A, poderá melhorar substancialmente o desempenho da consulta alternando para uma estrutura de célula de Voronoi diferente, principalmente se seu aplicativo executar várias consultas de polígono de interseção de polígono.

Dados Muito Densos
Se seus dados estiverem concentrados em regiões muito pequenas (por exemplo, você possui centenas de objetos por quilômetro quadrado), poderá melhorar o desempenho da consulta utilizando uma estrutura de célula de Voronoi cuja densidade da célula corresponda à densidade de dados.

Conceitos Relacionados:

- "Índices Geodésicos de Voronoi" na página 177
- "Estruturas de Células de Voronoi" na página 178

Tarefas Relacionadas:

- "Criando Índices Geodésicos de Voronoi" na página 181

Referência Relacionada:

- "Instrução CREATE INDEX para um Índice Geodésico de Voronoi" na página 182

• "Estruturas de Células de Voronoi Fornecidas com o DB2 Geodetic Extender" na página 184

Criando Índices Geodésicos de Voronoi

O DB2 Geodetic Extender fornece um novo método de acesso espacial que permite criar índices em colunas contendo dados geodésicos. As consultas que utilizam um índice podem ser executadas mais rapidamente.

Pré-requisitos:

Antes de criar um índice geodésico de Voronoi, seu ID do usuário deve conter as mesmas autorizações e privilégios de quando você cria um índice de grade espacial (consulte "Criando Índices de Grades Espaciais" na página 108).

Restrições:

As mesmas restrições para criar índices utilizando a instrução CREATE INDEX entram em vigor quando você cria um índice geodésico de Voronoi. Ou seja, a coluna na qual você cria um índice deve ser uma coluna de tabela básica, não uma coluna de exibição ou uma coluna de pseudônimo. O DB2 UDB resolverá aliases no processo.

Procedimento:

Você pode criar um índice geodésico de Voronoi de uma das seguintes formas:

- Utilize a janela Criar Índice do Centro de Controle do DB2.
- Utilize a instrução SQL CREATE INDEX com a extensão db2gse.spatial_index na cláusula EXTEND USING.

Para criar um índice geodésico de Voronoi utilizando o Centro de Controle:

1. No Centro de Controle, clique com o botão direito na tabela que possui a coluna espacial na qual você deseja criar um índice geodésico de Voronoi e selecione **Spatial Extender ⟶ Índices Espaciais** no menu pop-up. A janela Índices Espaciais aparece.
2. Siga as instruções na ajuda on-line para a janela Índices Espaciais. Você pode exibir estas instruções clicando no botão de **Ajuda** na janela Índices Espaciais.

Para criar um índice geodésico de Voronoi utilizando a instrução SQL CREATE INDEX:

Emita a instrução CREATE INDEX utilizando a cláusula EXTEND USING e a extensão de índice da grade db2gse.spatial_index.

Exemplo:

A instrução CREATE INDEX de exemplo a seguir cria o índice geodésico STORESX1 na coluna espacial LOCALIZAÇÃO na tabela CLIENTES:

```
CREATE INDEX storesx1
   ON customers (location)
   EXTEND USING db2gse.spatial_index (-1, -1, 1)
```

Para um índice geodésico de Voronoi, você deve especificar o valor –1 nos dois primeiros parâmetros da cláusula USING db2gse.spatial_index. Para obter detalhes, consulte "Instrução CREATE INDEX para um Índice Geodésico de Voronoi" na página 182.

Conceitos Relacionados:

- "Índices Geodésicos de Voronoi" na página 177
- "Estruturas de Células de Voronoi" na página 178
- "Considerações para Seleção de uma Estrutura de Célula de Voronoi Alternativa" na página 179

Tarefas Relacionadas:

- "Criando Índices de Grades Espaciais" na página 108

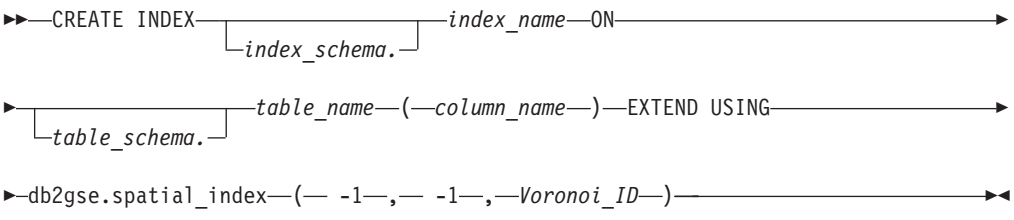
Referência Relacionada:

- "Instrução CREATE INDEX para um Índice Geodésico de Voronoi" na página 182
- "Estruturas de Células de Voronoi Fornecidas com o DB2 Geodetic Extender" na página 184
- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

Instrução CREATE INDEX para um Índice Geodésico de Voronoi

Utilize a instrução CREATE INDEX com a cláusula EXTEND USING para criar um índice geodésico de Voronoi.

Sintaxe:



Em que:

index_schema
: Nome do esquema ao qual deve pertencer o índice que está sendo criado. Se você não especificar um nome, o DB2 UDB utilizará o nome do esquema armazenado no registro especial CURRENT SCHEMA.

index_name
: Nome não qualificado do índice geodésico que está sendo criado.

table_schema
: Nome do esquema ao qual pertence a tabela que contém *column_name*. Se você não especificar um nome, o DB2 UDB utilizará o nome do esquema armazenado no registro especial CURRENT SCHEMA.

table_name
: Nome não qualificado da tabela que contém *column_name*.

column_name
: Nome da coluna espacial na qual o índice geodésico de Voronoi será criado.

Voronoi_ID
: Um inteiro que identifica o ID da estrutura de célula de Voronoi. Estão disponíveis quatorze estruturas de célula de Voronoi. Um ID de Voronoi 1

especifica a estrutura de célula de Voronoi que é baseada na densidade populacional mundial que também é utilizada para todas as operações espaciais pelo DB2 Geodetic Extender.

Exemplos:

A instrução CREATE INDEX de exemplo a seguir cria o índice geodésico STORESX1 na coluna espacial LOCALIZAÇÃO na tabela CLIENTES:

```
CREATE INDEX storesx1
   ON customers (location)
   EXTEND USING db2gse.spatial_index (-1, -1, 1)
```

O otimizador considera um índice de Voronoi para utilização por todas as consultas que contêm as seguintes funções em sua cláusula WHERE:

- ST_Contains
- ST_Distance
- ST_Intersects
- ST_MBRIntersects
- ST_EnvIntersects
- EnvelopesIntersect
- ST_Within

As instruções a seguir demonstram a utilização de um índice de Voronoi. Primeiro, insira dados na tabela CLIENTE. Você pode inserir valores diretamente, conforme mostrado nesta primeira instrução INSERT:

```
INSERT INTO customer
(id, last_name, first_name, address, city, state, zip,
location)
  VALUES
('123-456789', 'Duck', 'Donald',
'123 Mallard Way', 'Wetland Marsh', 'ND', '55555-5555',
db2gse.ST_GeomFromWKT('POINT(123.123, 45.67)', 2000000000))
```

Como alternativa, você pode utilizar variáveis em um aplicativo, conforme mostra a próxima consulta, para inserir valores em uma tabela:

```
INSERT INTO customer
(id, last_name, first_name,
address, city, state, zip,
location)
  VALUES
(:mid, :mlast, :mfirst,
:maddress, :mcity, :mstate, :mzip,
db2gse.ST_GeomFromWKB(:mlocation))
```

A instrução UPDATE a seguir modifica os dados inseridos. Ela não utiliza o índice STORESX1 porque não utiliza a função ST_Contains, ST_Distance, ST_Intersects, ST_MBRIntersects, ST_EnvIntersects, EnvelopesIntersect ou ST_Within em sua cláusula WHERE.

```
UPDATE customer
SET location = db2gse.ST_GeomFromWKT('POINT(123.123, 45.67)',
2000000000)
WHERE id = '123-456789';
```

As seguintes instruções DELETE podem utilizar o índice STORESX1, se o otimizador determinar que o índice melhora o desempenho porque as instruções DELETE utilizam a função ST_Within e as funções ST_Intersects em suas cláusulas WHERE, respectivamente:

```
DELETE FROM customers
WHERE db2gse.ST_Within(location, :BayArea) = 1;
DELETE FROM customers
WHERE db2gse.ST_Intersects(c.location, :BayArea) = 1
```

As duas instruções SELECT a seguir também podem utilizar o índice STORESX1:

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

Conceitos Relacionados:

- "Índices Geodésicos de Voronoi" na página 177
- "Estruturas de Células de Voronoi" na página 178
- "Considerações para Seleção de uma Estrutura de Célula de Voronoi Alternativa" na página 179

Tarefas Relacionadas:

- "Criando Índices Geodésicos de Voronoi" na página 181

Referência Relacionada:

- "Estruturas de Células de Voronoi Fornecidas com o DB2 Geodetic Extender" na página 184

Estruturas de Células de Voronoi Fornecidas com o DB2 Geodetic Extender

Cada estrutura de célula de Voronoi cobre toda a Terra. Nas ilustrações a seguir, apenas as partes da Terra na área em que as células são densas para esta estrutura de célula de Voronoi são mostradas. Quando selecionar uma estrutura de célula de Voronoi, lembre-se de que as células fora das áreas ilustradas serão grandes, com resolução equivalentemente inferior. Se seus dados estiverem localizados nestas áreas esparsas, o desempenho da consulta poderá ser prejudicado.

A tabela a seguir lista as estruturas de células de Voronoi fornecidas pelo DB2 Geodetic Extender. Estas estruturas de células de Voronoi são fornecidas pela Geodyssey Ltd.

Tabela 25. Estruturas de Células de Voronoi

Descrição	ID de Voronoi	Ilustração
Mundo, com base na densidade populacional	1	Figura 22 na página 185
Estados Unidos	2	Figura 23 na página 186
Canadá	3	Figura 24 na página 187
Índia	4	Figura 25 na página 188
Japão	5	Figura 26 na página 189
África	6	Figura 27 na página 190
Austrália	7	Figura 28 na página 191
Europa	8	Figura 29 na página 192

Tabela 25. Estruturas de Células de Voronoi (continuação)

Descrição	ID de Voronoi	Ilustração
América do Norte	9	Figura 30 na página 193
América do Sul	10	Figura 31 na página 194
Mediterrâneo	11	Figura 32 na página 195
Mundo, distribuição de dados uniforme, resolução média (dodeca04)	12	Figura 33 na página 196
Mundo, com base na saída industrial (países do G7)	13	Figura 34 na página 197
Mundo, distribuição de dados uniforme, resolução baixa (isotype)	14	Figura 35 na página 198

Mundo, com Base na Densidade Populacional (ID de Voronoi: 1)

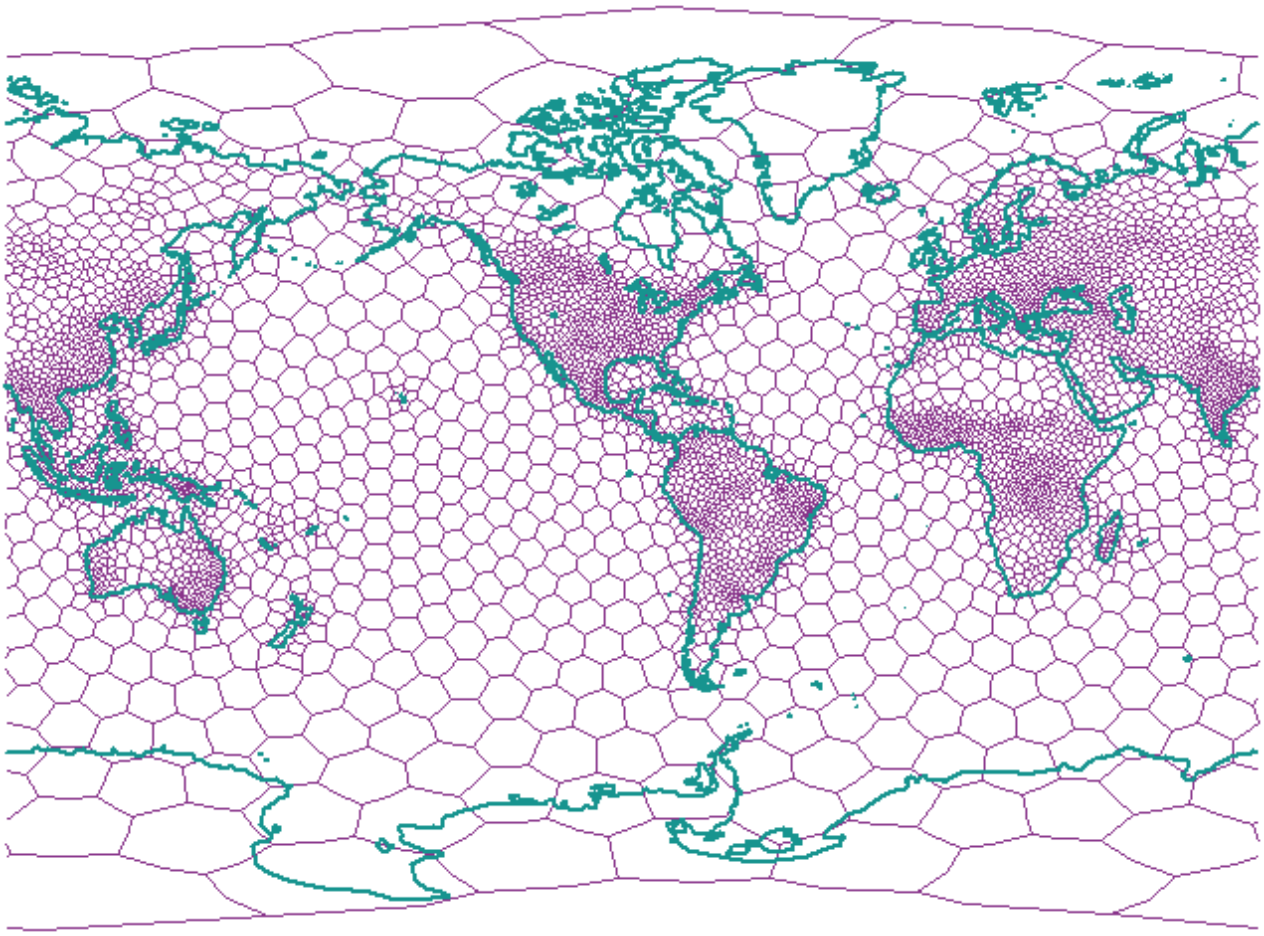


Figura 22. Estrutura de Célula de Voronoi para o Mundo (População)

Estados Unidos (ID de Voronoi: 2)

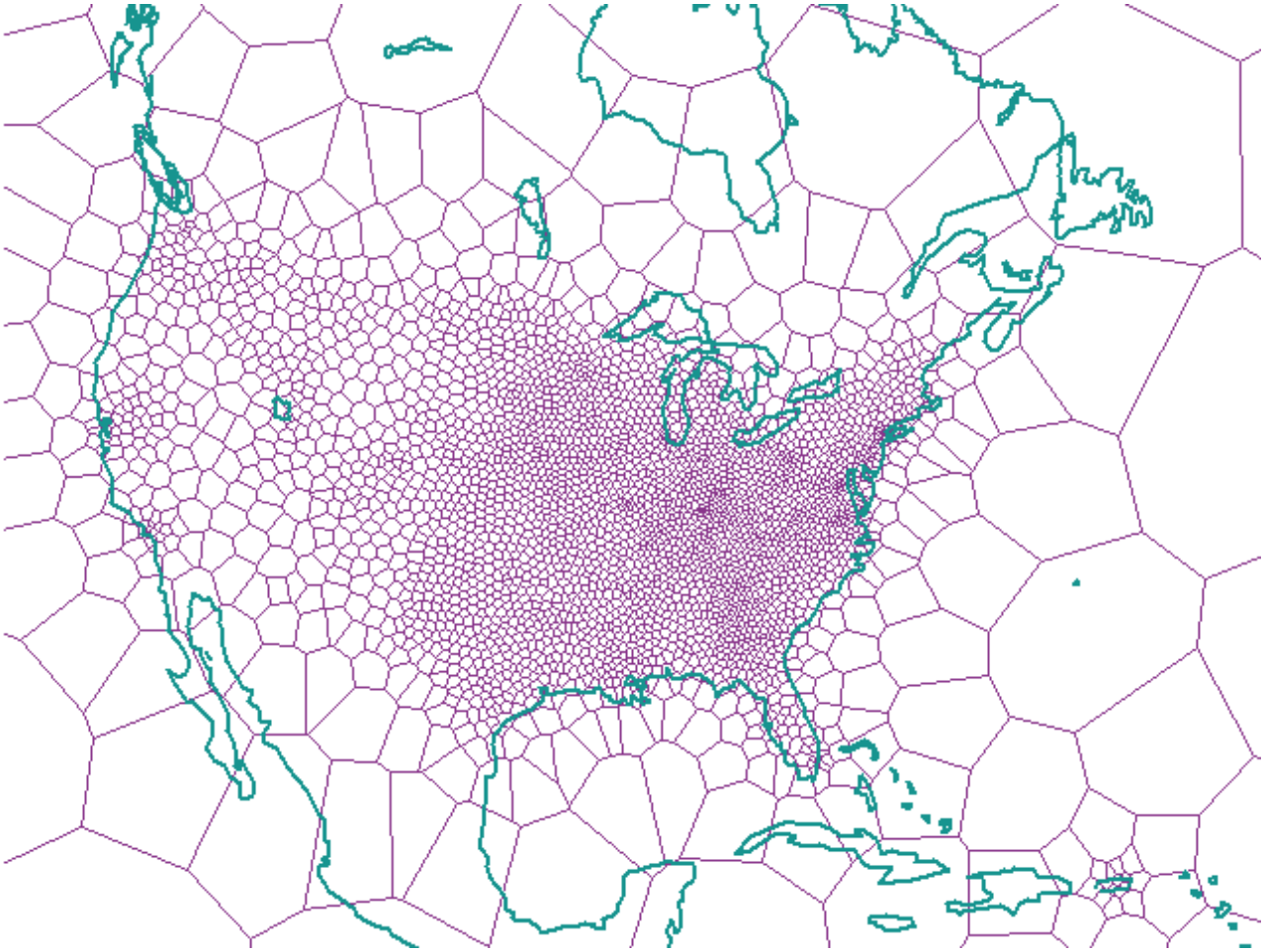


Figura 23. Estrutura de Célula de Voronoi para os E.U.A.

Canadá (ID de Voronoi: 3)

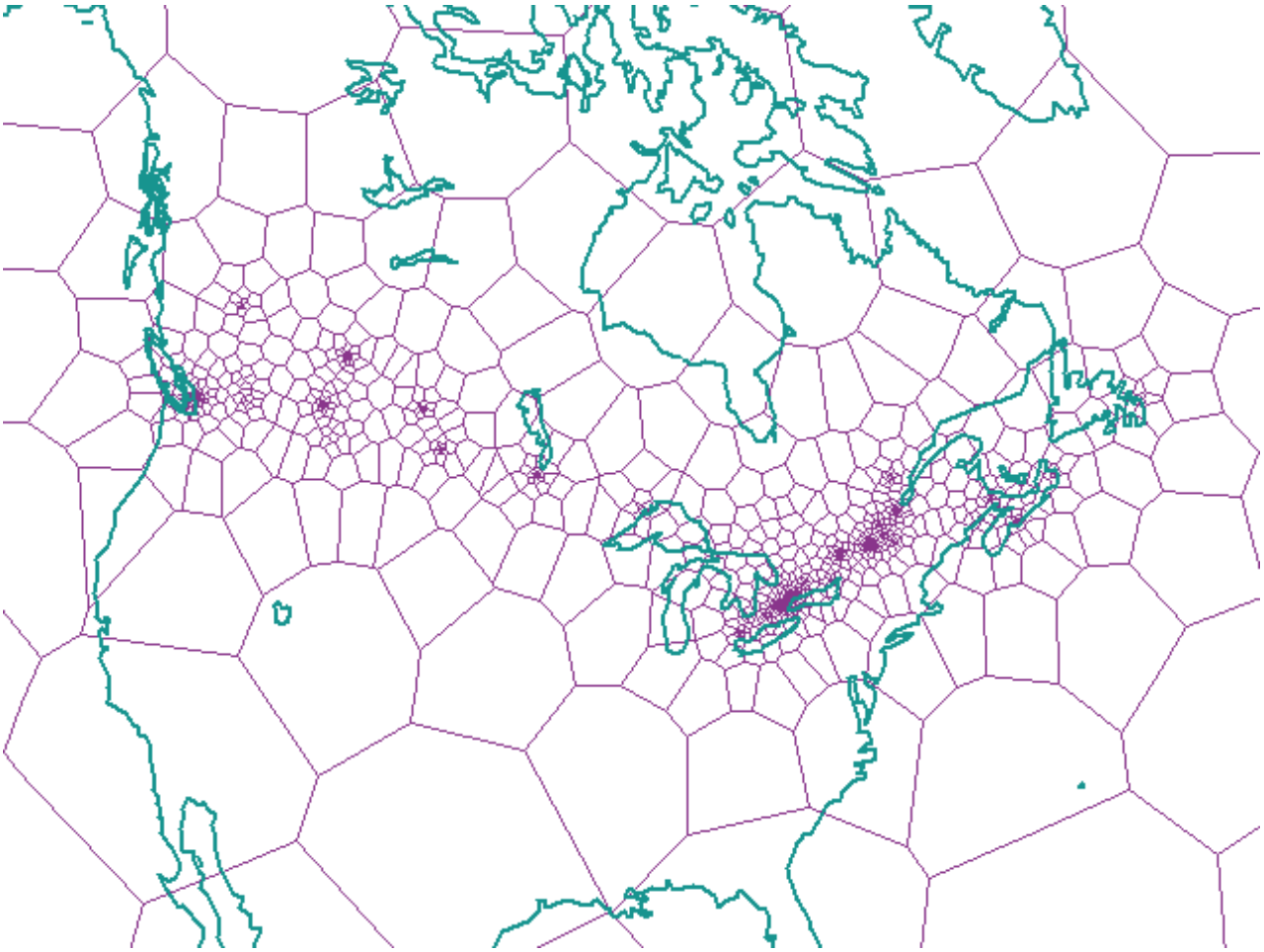


Figura 24. Estrutura de Célula de Voronoi para o Canadá

Índia (ID de Voronoi: 4)

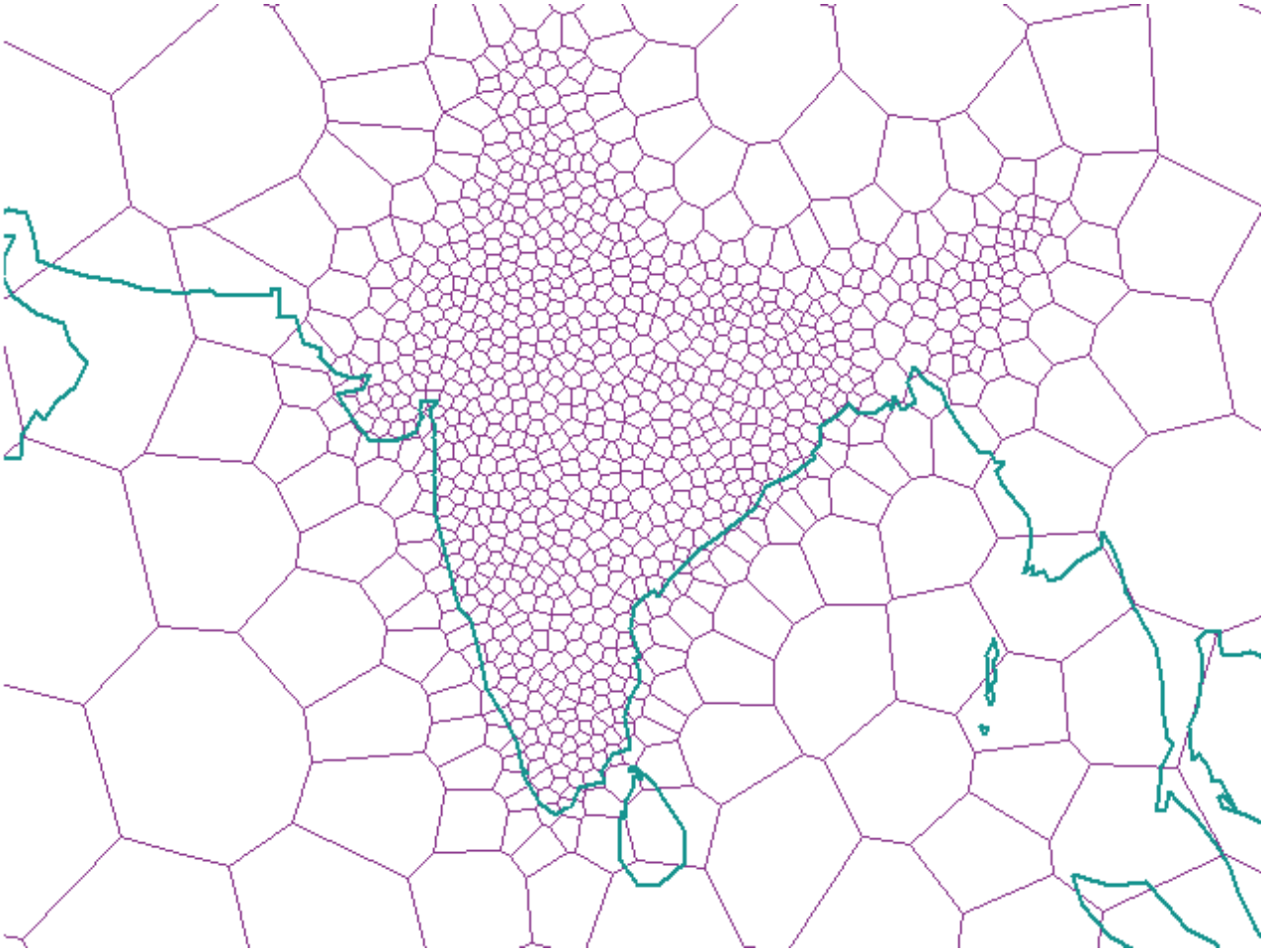


Figura 25. Estrutura de Célula de Voronoi para a Índia

Japão (ID de Voronoi: 5)

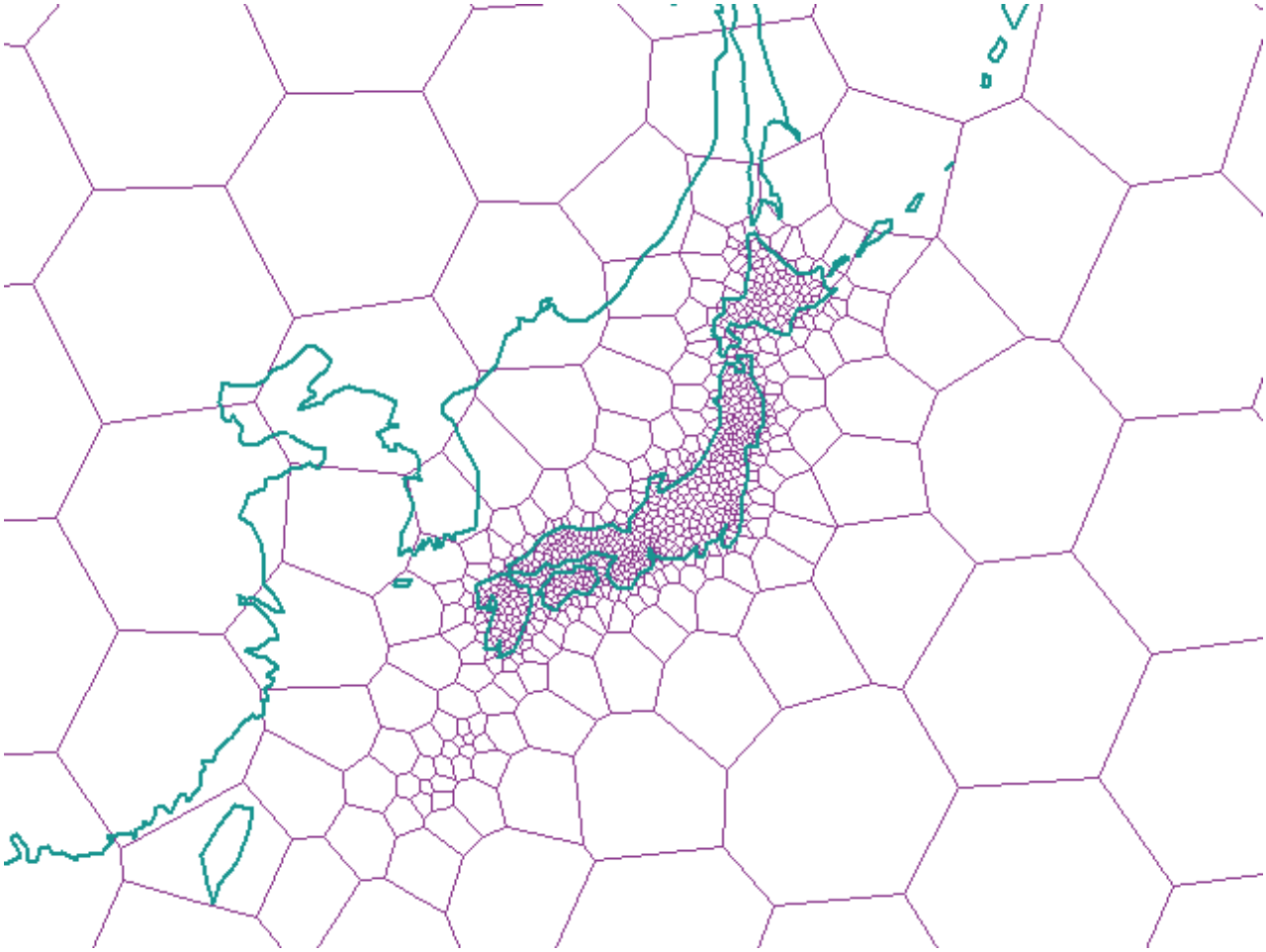


Figura 26. Estrutura de Célula de Voronoi para o Japão

África (ID de Voronoi: 6)

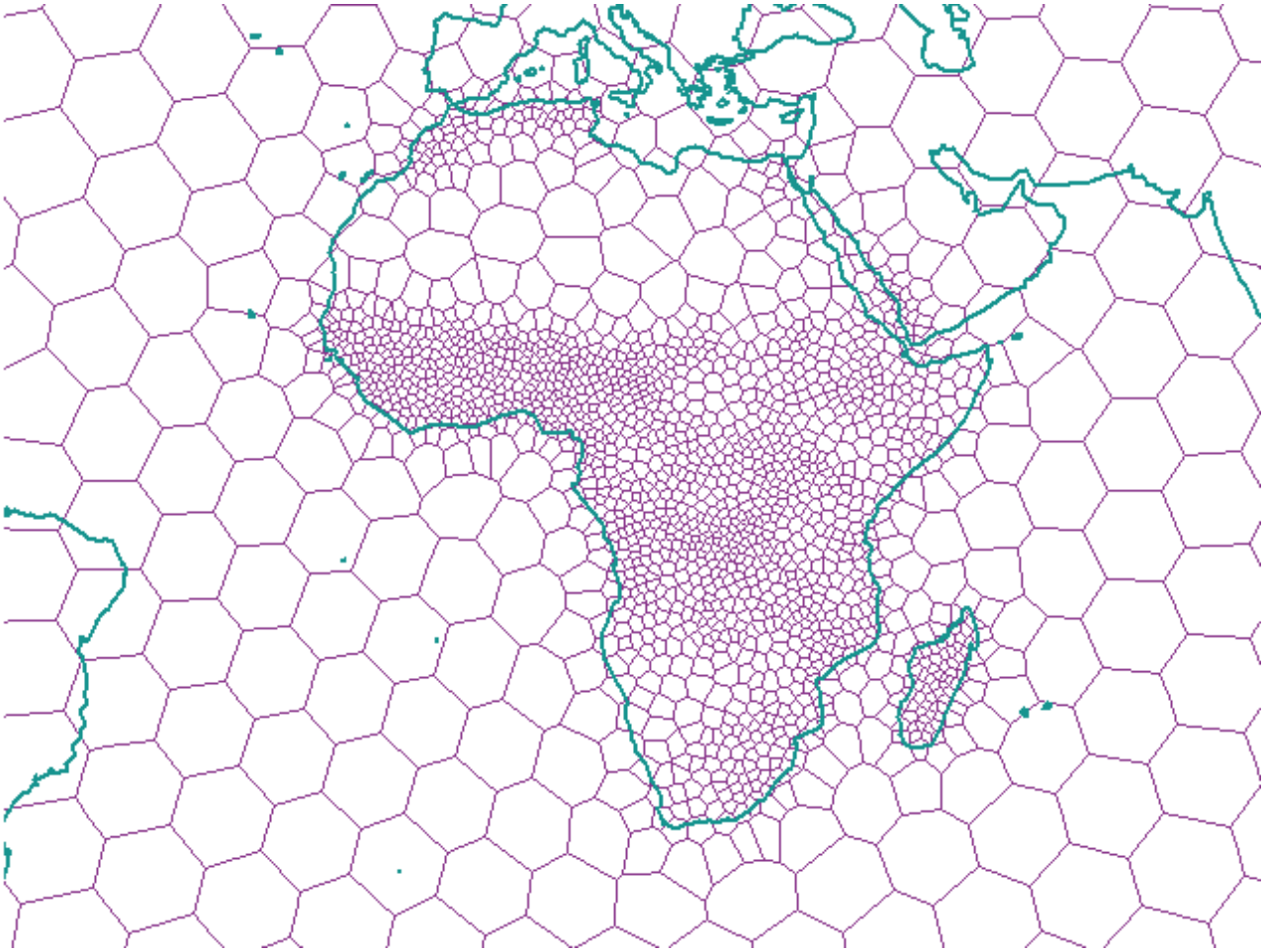


Figura 27. Estrutura de Célula de Voronoi para a África

Austrália (ID de Voronoi: 7)

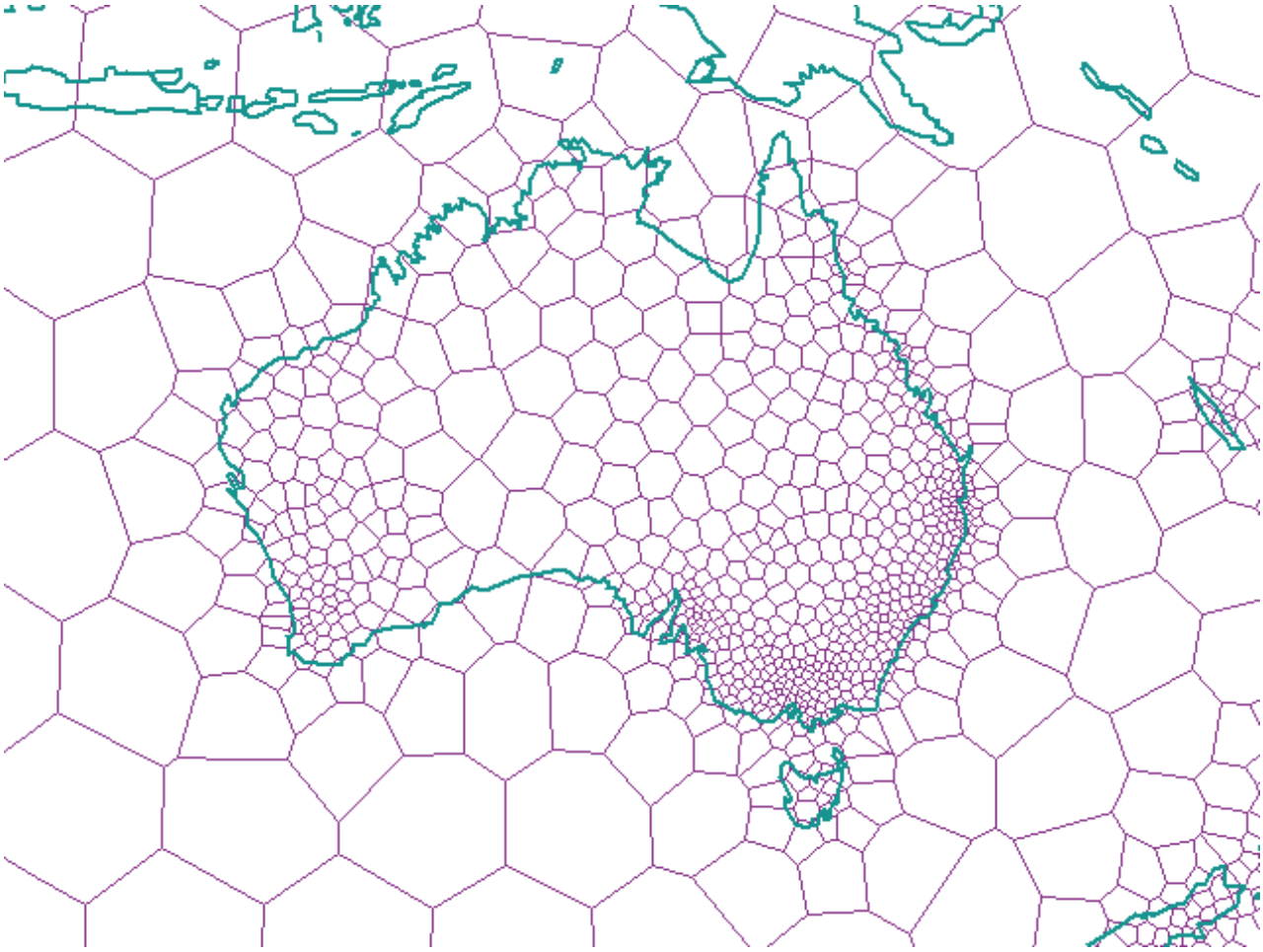


Figura 28. Estrutura de Célula de Voronoi para a Austrália

Europa (ID de Voronoi: 8)

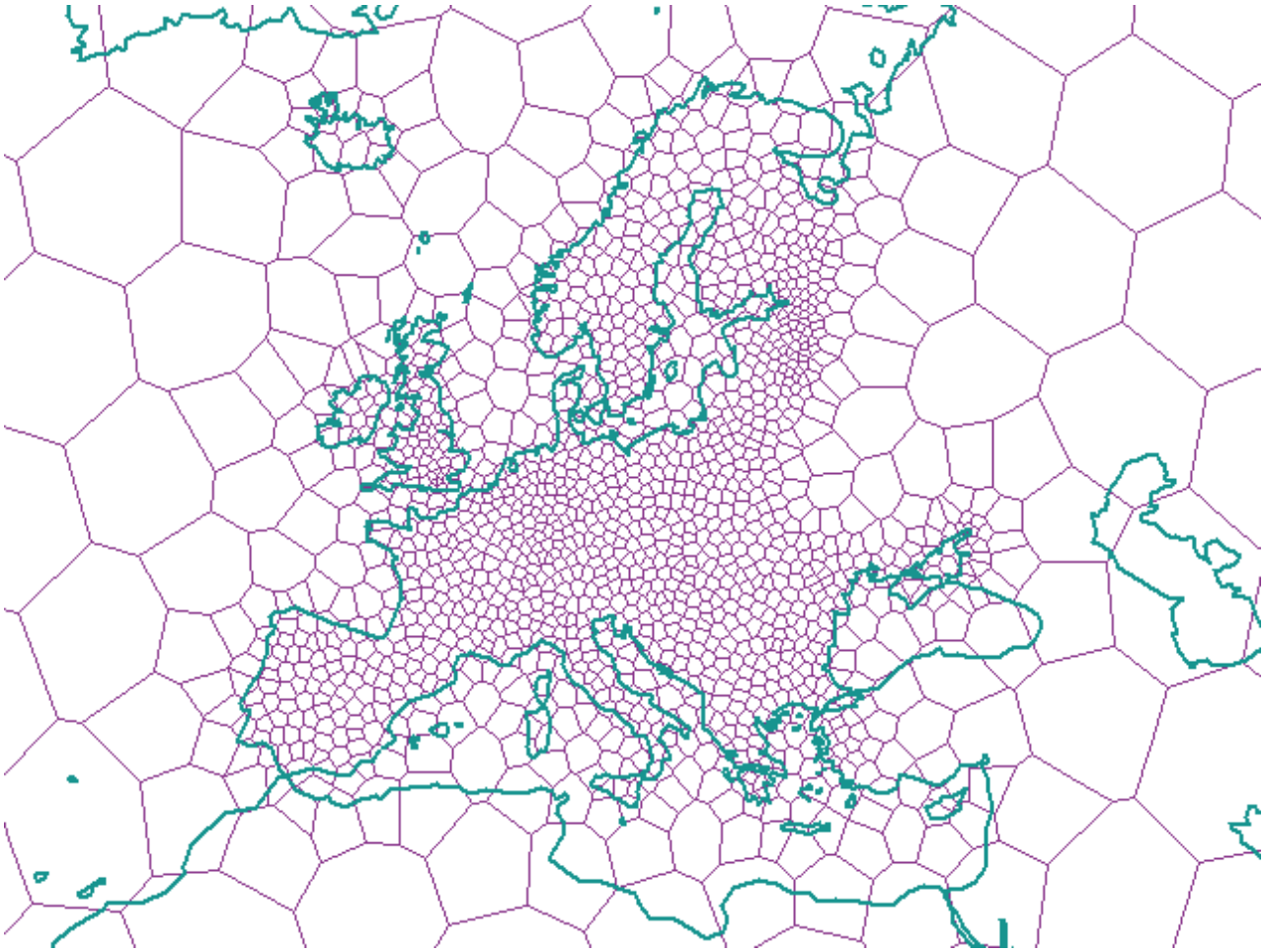


Figura 29. Estrutura de Célula de Voronoi para a Europa

América do Norte (ID de Voronoi: 9)

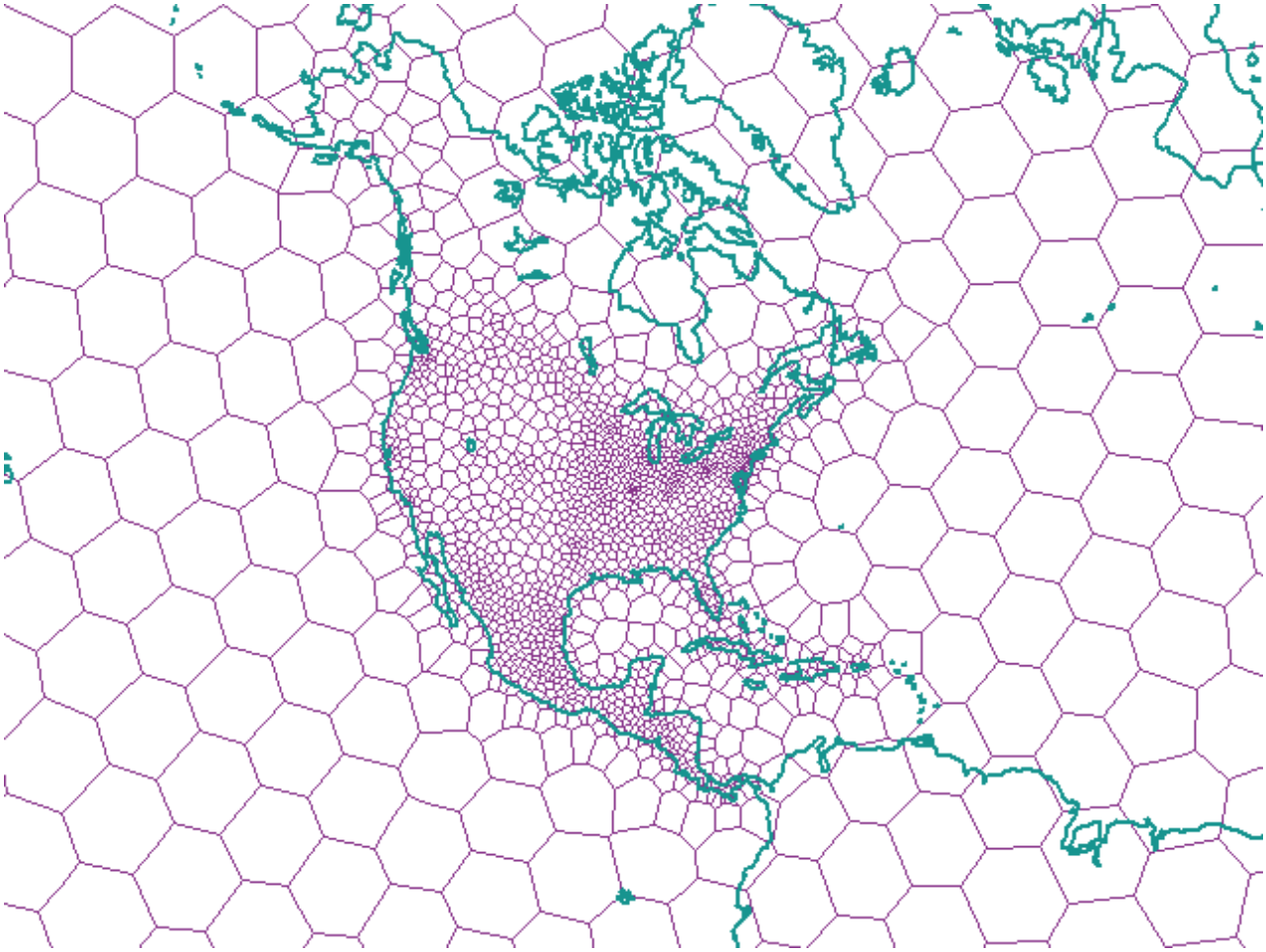


Figura 30. Estrutura de Célula de Voronoi para a América do Norte

América do Sul (ID de Voronoi: 10)

Figura 31. Estrutura de Célula de Voronoi para a América do Sul

Mediterrâneo (ID de Voronoi: 11)

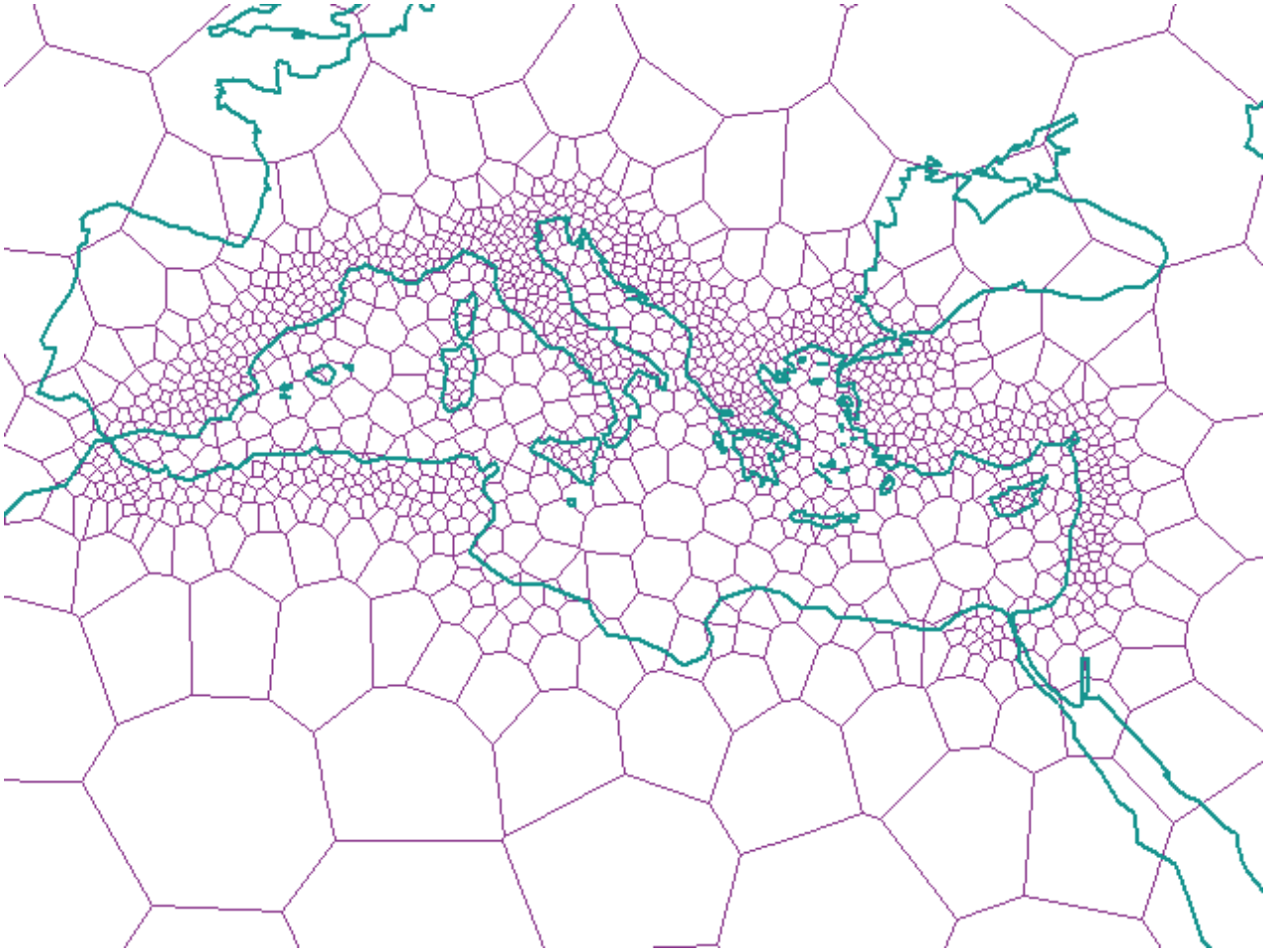


Figura 32. Estrutura de Célula de Voronoi para a Área do Mediterrâneo

Mundo, Distribuição de Dados Uniforme, Resolução Média – dodeca04 (ID de Voronoi: 12)

Figura 33. Estrutura de Célula de Voronoi para o Mundo (dodeca04)

Mundo, Nações Industriais – Países do G7 (ID de Voronoi: 13)

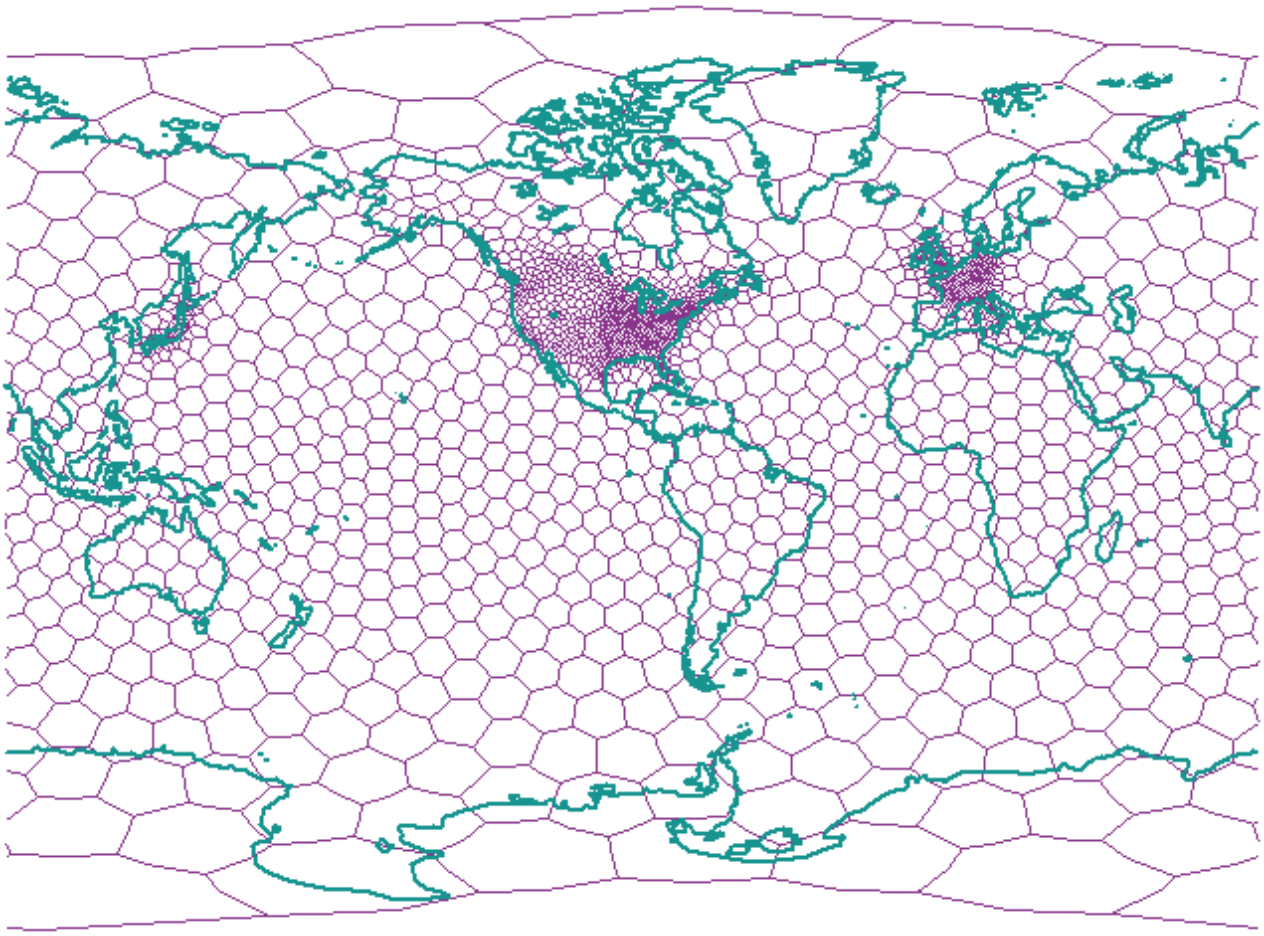


Figura 34. Estrutura de Célula de Voronoi para (g7nations)

Mundo, Distribuição de Dados Uniforme, Resolução Baixa – isotype (ID de Voronoi: 14)

Figura 35. Estrutura de Célula de Voronoi para o Mundo (isotype)

Capítulo 19. Diferenças ao Utilizar Dados Geodésicos e Espaciais

Este capítulo descreve as seguintes diferenças ao utilizar dados geodésicos e espaciais:

- Atributos x e y mínimos e máximos para os tipos de dados ST_Geometry
- Diferenças ao trabalhar com representações plana e redonda da Terra
- Funções espaciais suportadas pelo DB2 Geodetic Extender e diferenças no comportamento das funções
- Procedimentos armazenados e exibições de catálogos suportados pelo DB2 Geodetic Extender
- Sistemas de referências espaciais geodésicos adicionais (datums) e elipsóides geodésicos

Atributos X e Y Mínimo e Máximo

O DB2® Geodetic Extender utiliza um MBC (Minimum Bounding Circle) em vez de um retângulo de limite mínimo para organizar dados em estruturas de células para o índice geodésico de Voronoi.

Para geometrias geodésicas, o MBC é um círculo que cerca as geometrias e o X e y mínimos e máximos possuem os seguintes valores internos:

xmin O termo *i* do co-seno de direção do centro do círculo de limite.

xmax O termo *j* do co-seno de direção do centro do círculo de limite.

ymin O termo *k* do co-seno de direção do centro do círculo de limite.

ymax O *arc_radius* do círculo de limite.

Para geometrias geodésicas, as funções ST_MinX, ST_MaxX, ST_MinY e ST_MaxY exibem pontos no MBC. Os resultados destas funções ainda produzem valores de longitude e latitude semelhantes a geometrias espaciais, mas os resultados podem ser diferentes para geometrias geodésicas, conforme a seguir:

- Se o MBC cruzar o meridiano de data, o valor ST_MinX será maior que o valor ST_MaxX.Por exemplo, se o centro de um MBC estiver no meridiano de data e tiver um raio de 5 graus, o valor ST_MinX será 175 e o valor ST_MaxX será –175.
- Se o MBC incluir o Pólo Norte ou o Pólo Sul, ST_MinX será –180 e ST_MaxX será 180.
- Se o MBC incluir o Pólo Norte, o valor ST_MaxY será 90.
- Se o MBC incluir o Pólo Sul, o valor ST_MinY será –90.

Diferenças em Trabalhar com Representações Planas e Redondas da Terra

O DB2 Spatial Extender e o DB2 Geodetic Extender utilizam tecnologias principais diferentes:

- O Spatial Extender utiliza um mapa plano (ou planar), com base nas coordenadas projetadas. No entanto, nenhuma projeção de mapa pode representar fielmente toda a Terra porque todos os mapas possuem bordas; enquanto a Terra não possui.
- O Geodetic Extender utiliza um elipsóide como seu modelo para tratar a Terra como um globo total, sem distorções nos pólos ou bordas no meridiano de 180 graus.

Nesta seção, o termo ″Terra plana″ refere-se à utilização de uma projeção para representar a Terra inteira. O termo ″Terra redonda″ refere-se à utilização de um sistema de referência que utiliza um elipsóide como seu modelo de Terra.

As diferentes tecnologias geram diferenças em como as geometrias são tratadas em algumas situações, principalmente as ilustradas neste tópico:

- Segmentos lineares (e distâncias medidas) que cruzam o meridiano de 180 graus.
- Polígonos que estendem o meridiano de 180 graus.
- Retângulos mínimos de limites que cruzam o meridiano de 180 graus.
- Polígonos que incluem um pólo.
- Polígonos que representam hemisférios, faixas equatoriais ou toda a Terra.

O Geodetic Extender possui vantagens específicas quando você está trabalhando com geometrias que cruzam o meridiano de 180 graus ou estão próximas a um pólo, onde a representação plana da Terra utilizada pelo Spatial Extender encontra limitações.

Segmentos Lineares que Cruzam o Meridiano de 180 Graus

A Figura 36 na página 201 mostra as diferentes formas que o Spatial Extender e o Geodetic Extender tratam um segmento linear que cruza o meridiano de 180 graus. Neste exemplo, o segmento linear é utilizado para calcular a distância entre Anchorage e Tóquio. O Geodetic Extender calcula a distância entre dois pontos em um geodésico, o caminho mais curto entre dois pontos no elipsóide (consulte “Distâncias Geodésicas” na página 162). Os dois pontos podem estar localizados em qualquer lugar no globo, e o Geodetic Extender escolhe corretamente um segmento linear que percorre de Anchorage a Tóquio, porque ele utiliza a representação redonda da Terra. No entanto, como o Spatial Extender utiliza a projeção de mapa plano, o Spatial Extender não sabe que um segmento linear pode ligar Anchorage e Tóquio dessa forma e escolhe um segmento linear muito mais longo que percorre da direção leste até Tóquio. A projeção do mapa plano tem o meridiano de –180 graus na borda esquerda e o meridiano de 180 graus na borda direita.

Para obter um resultado correto utilizando o Spatial Extender, é necessário executar uma das seguintes ações:

- Divida o segmento linear em dois, um a leste do meridiano de 180 graus e o outro a oeste dele.
- Reprojete os dados de forma que o meridiano de 180 graus não fique em uma borda.

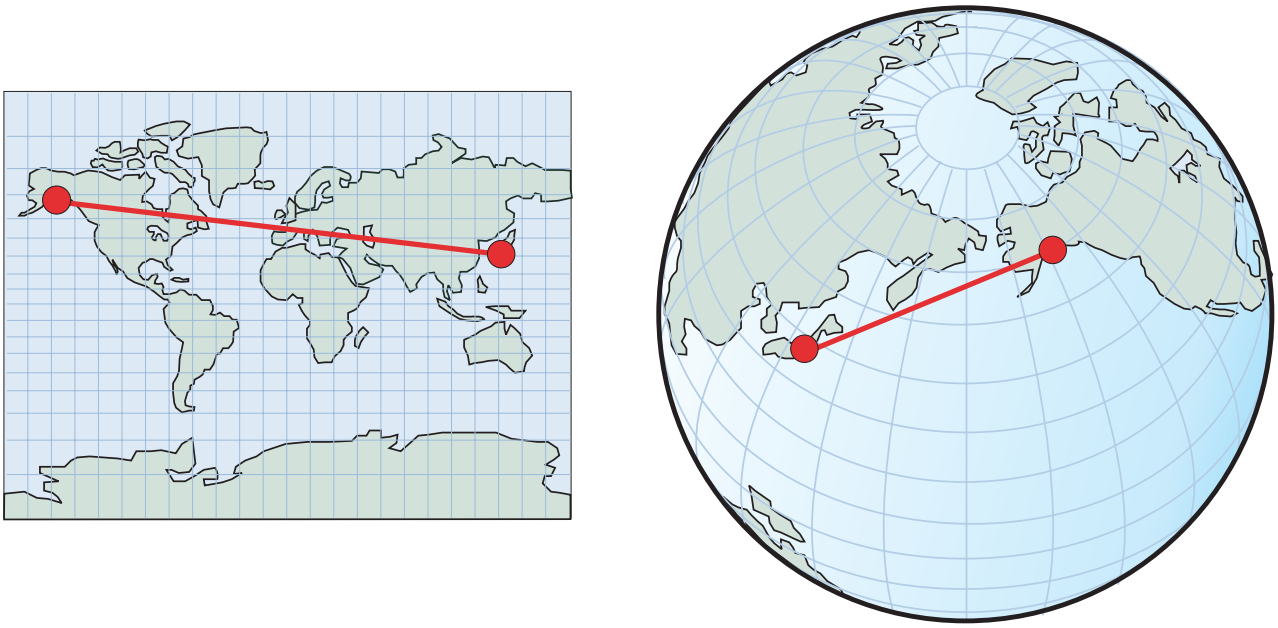


Figura 36. Linhas que Cruzam o Meridiano de 180 Graus

Polígonos que Estendem o Meridiano de 180 Graus

Para tratar um polígono que estende o meridiano de 180 graus, a representação plana da Terra (Spatial Extender) requer que você divida o polígono em duas partes, um polígono para a parte a leste do meridiano de 180 graus e um polígono para a parte a oeste do meridiano:

```
  MULTIPOLYGON(
((–180 30, –165 30, –165 40, –180 40, –180 30)),
((180 30, 180 40, 165 40, 165 30, 180 30)))
```

Conforme mostra a Figura 37 na página 202, a representação redonda da Terra (Geodetic Extender) não requer esta divisão e você pode utilizar um único polígono inalterado:

```
  POLYGON((165 30, –165 30, –165 40, 165 40, 165 30))
```

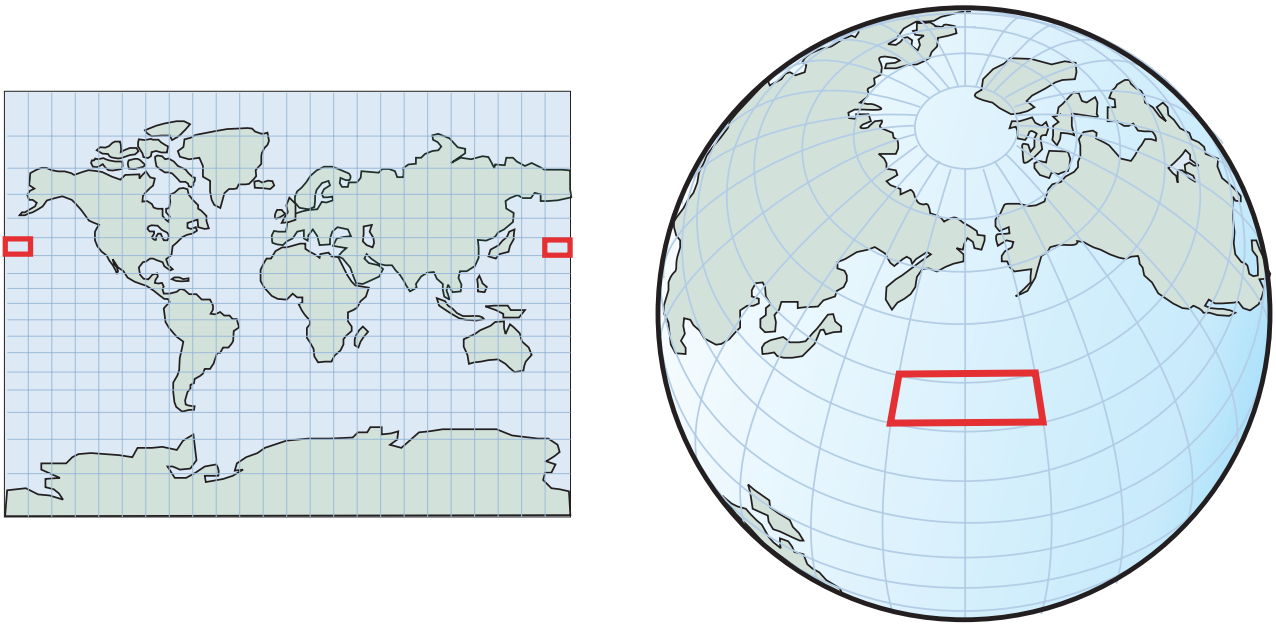


Figura 37. Polígonos que Estendem o Meridiano de 180 Graus—Criar Dois Polígonos Separados

Se você não criou dois polígonos separados ao utilizar o Spatial Extender, ele realmente reordenará os vértices do polígono para que seja definida uma área diferente, conforme mostra a Figura 38 na página 203. A parte superior da Figura 38 na página 203 mostra os vértices corretos de um polígono estendendo o meridiano de 180 graus:

```
POLYGON((90 0, -90 0, -90 40, 90 40, 90 0))
```

A parte inferior da Figura 38 na página 203 mostra os vértices reordenados, que resultam em um polígono que não mais estende o meridiano de 180 graus, mas agora estende o meridiano de 0 grau.

```
POLYGON((-90 0, 90 0, 90 40, -90 40, -90 0))
```

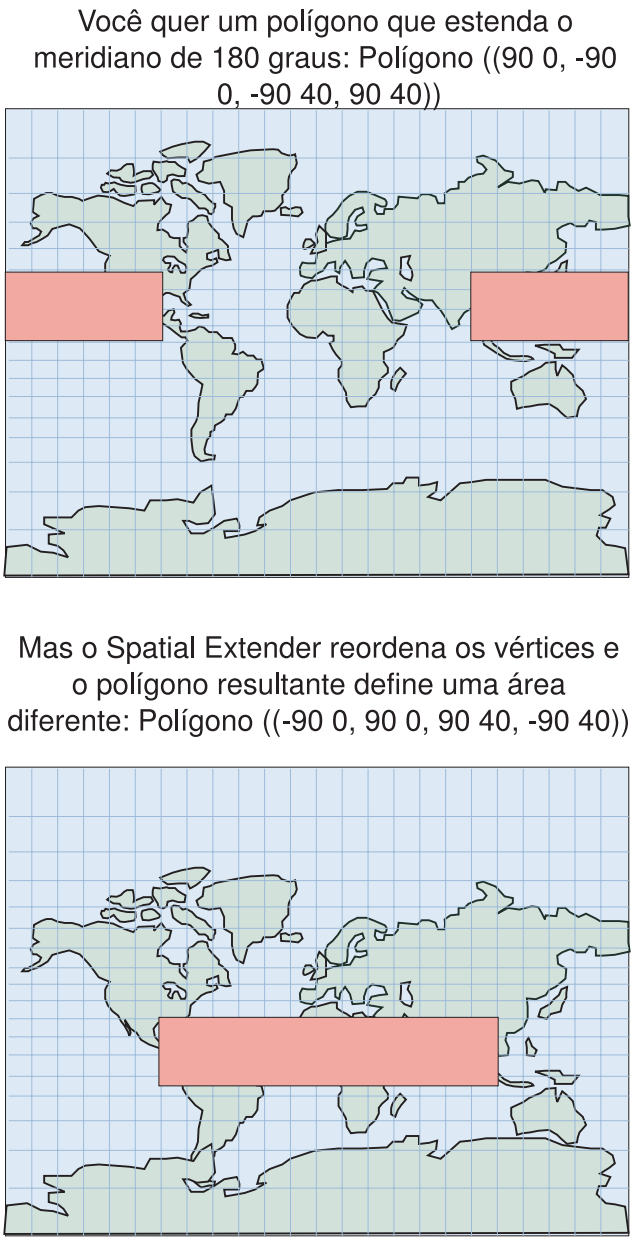


Figura 38. Polígonos que Estendem o Meridiano de 180 Graus—Vértices Reordenados

A área definida seria a área complementar da Terra e não a área pretendida, conforme mostrado em Figura 39 na página 204. Semelhante ao exemplo de segmento linear acima, outra forma de tratar esta situação é reprojetar os dados de forma que o meridiano de 180 graus não fique em uma borda.

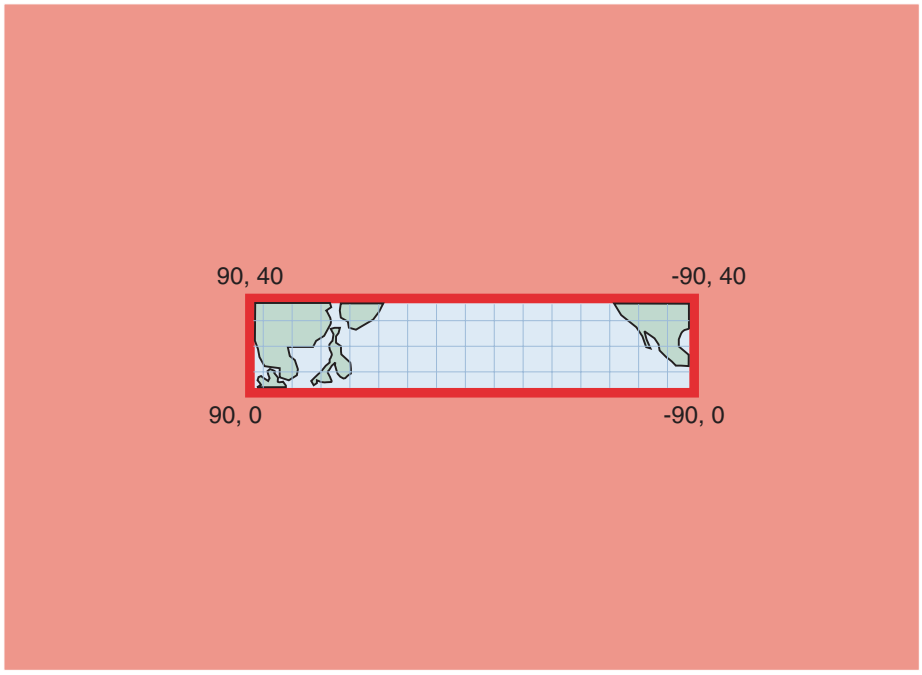


Figura 39. Polígonos que Estendem o Meridiano de 180 Graus—Área Complementar

Polígonos que Incluem um Pólo

A Figura 40 na página 205 mostra como você pode trabalhar com um polígono que inclui o Pólo Sul com o Spatial Extender ou com o Geodetic Extender. Como você está trabalhando diretamente na borda da projeção de mapa plano com o Spatial Extender, a distorção do mapa da superfície da Terra requer a adição de bordas e vértices para representar o pólo em um polígono:

```
POLYGON((–180 –90, 180 –90, 180 –60, –180 –60, –180 –90))
```

A representação redonda da Terra (Geodetic Extender) mostra o polígono em torno do Pólo Norte como um círculo que segue o paralelo Sul de –60°:

```
POLYGON((0 –60, –1 –60, –2 –60, ..., –179 –60, 180 –60, 179 –60, ..., 1 –60, 0 –60))
```

Uma melhor forma de representar este círculo é reprojetar os dados de forma que todo o Pólo Sul e a área circundante fiquem visíveis no mapa.

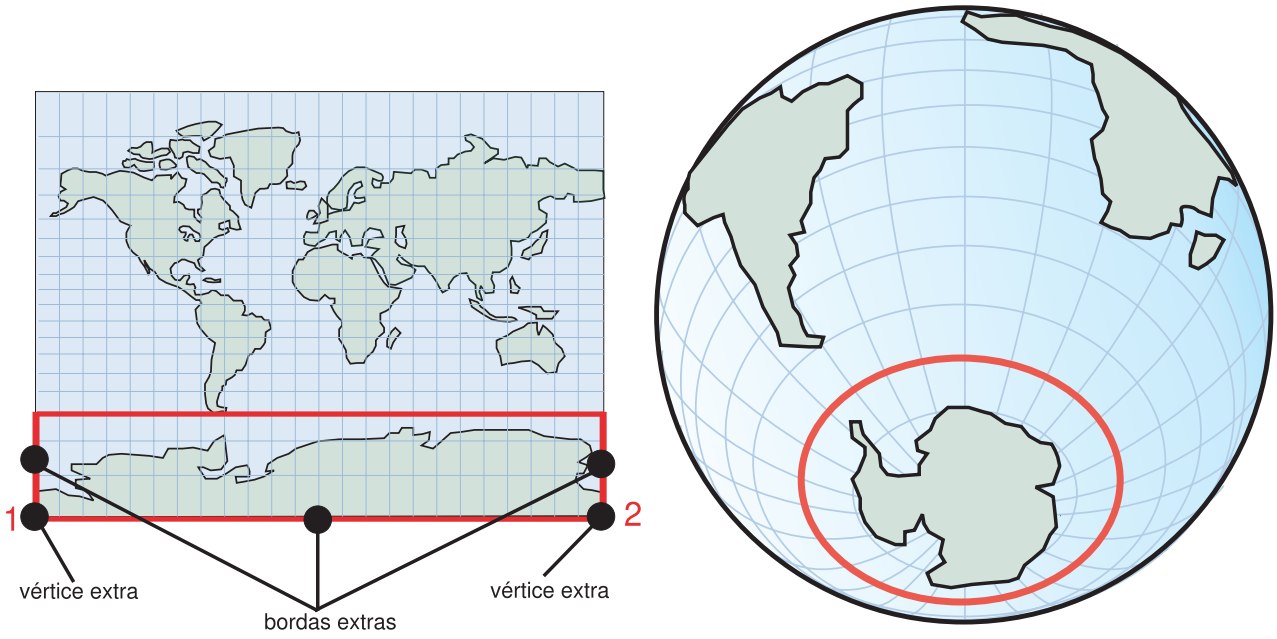


Figura 40. Polígonos que Incluem um Pólo

Nos exemplos acima, você pode obter resultados precisos se escolher um sistema de referência espacial projetado apropriado. No entanto, nenhuma projeção poderá resolvê-los simultaneamente. Por exemplo, uma projeção que não possui o meridiano de 180 graus na borda coloca a borda em algum outro lugar e desloca a área do problema.

Polígonos que Representam Hemisférios, Faixas Equatoriais e Toda a Terra

Quando precisar utilizar um polígono para representar grandes áreas da superfície da Terra, como um dos hemisférios, as faixas equatoriais ou toda a própria Terra, esteja ciente das diferentes formas que o Spatial Extender e o Geodetic Extender tratam estes casos. Nestas situações, uma representação redonda da Terra obtém resultados precisos para cálculos de distância e área, enquanto uma opção de projeção cuidadosa não pode obter.

Por exemplo, a Figura 41 na página 206 mostra os polígonos que definem o hemisfério Ocidental em uma representação plana da Terra (Spatial Extender) e em uma representação redonda da Terra (Geodetic Extender).

- Na representação plana da Terra na parte superior da Figura 41 na página 206, quatro coordenadas representam o hemisfério Ocidental em formato de texto bem conhecido como 'POLYGON((0 -90, 0 90, -180 90, 180 -90, 0 -90))'.
- Na representação redonda da Terra, quatro coordenadas representam o hemisfério Ocidental em formato de texto bem conhecido como 'POLYGON((0 0, 0 90, 180 0, 0 -90, 0 0))'. Estas quatro coordenadas definem um anel em volta da Terra no meridiano de 0 grau e sua linha antipodal, o meridiano de 180 graus.

Quando especificar os mesmos quatro pontos na ordem oposta, você define o hemisfério Oriental:

- Em uma representação plana da Terra, o hemisfério Oriental é 'POLYGON((0 -90, 180 -90, 180 90, 0 90, 0 -90))'.

- Em uma representação redonda da Terra, o hemisfério Oriental é 'POLYGON((0 -90, 180 0, 0 90, 0 0, 0 -90))'.

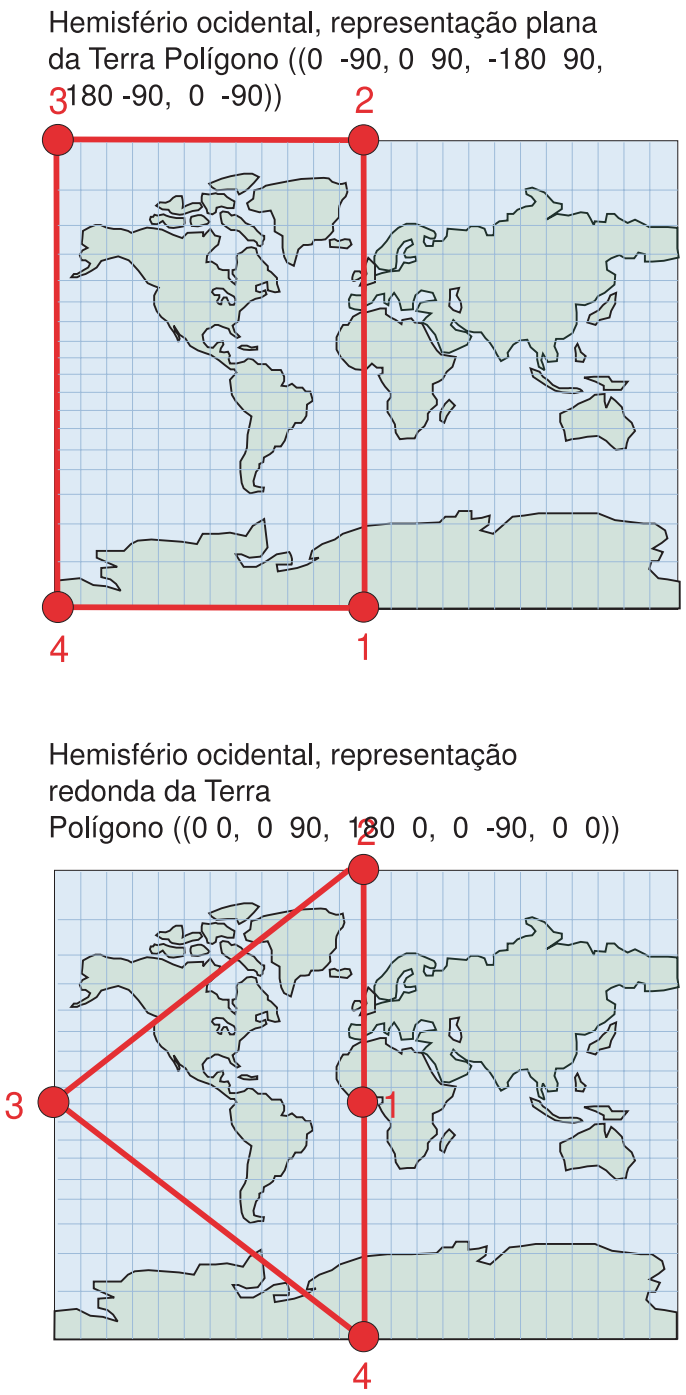


Figura 41. Polígonos que Representam Hemisférios

A Figura 42 na página 207 mostra as coordenadas de polígonos que definem a faixa equatorial em uma representação plana da Terra (Spatial Extender) e em uma representação redonda da Terra (Geodetic Extender).

- A parte superior da Figura 42 mostra a representação plana da Terra da faixa equatorial com coordenadas em formato de texto bem conhecido como 'POLYGON((180 –60, 180 60, –180 60, –180 –60, 180 –60))'.
- Na representação redonda da Terra na parte inferior da Figura 42, você define a área de exclusão de dois anéis para representar a faixa equatorial:

```
'MULTIPOLYGON(((0 60, -120 60, 120 60, 0 60)),
   ((0 -60, 120 -60, -120 -60, 0 -60)))'
```

São mostrados apenas três pontos em cada anel para esclarecimento. De fato, se você desejar que os anéis sigam mais de perto a linha de latitude de 60 graus ou de –60 graus, será necessário adicionar pontos mais intermediários. O primeiro anel ((0 60, -120 60, 120 60, 0 60)) especifica os vértices na ordem que define o sul da área da linha de latitude de 60 graus. O segundo anel ((0 -60, 120 -60, -120 -60, 0 -60)) especifica o norte da área da linha de latitude de –60 graus.

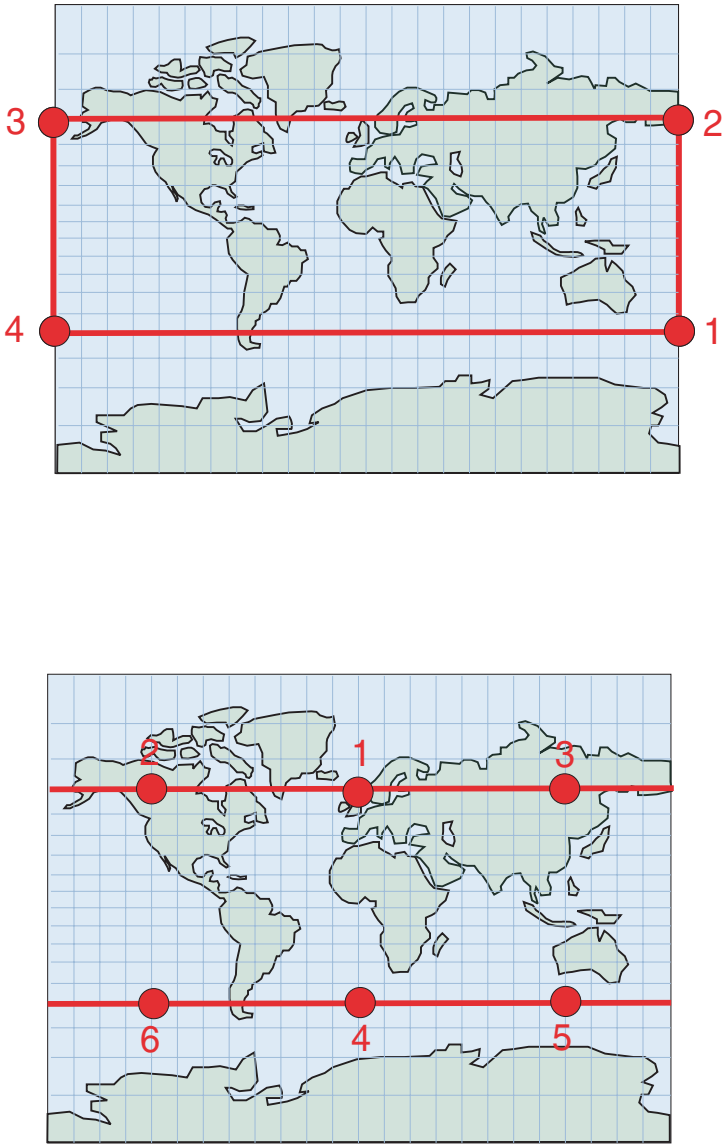


Figura 42. Polígonos que Representam Faixas Equatoriais

A Figura 43 mostra polígonos que definem toda a Terra em uma representação plana (Spatial Extender) e em uma representação redonda (Geodetic Extender). As duas representações representam toda a Terra com o mesmo polígono em formato de texto bem conhecido como 'POLYGON((-180 -90, 180 -90, 180 90, -180 90, -180 -90))'.

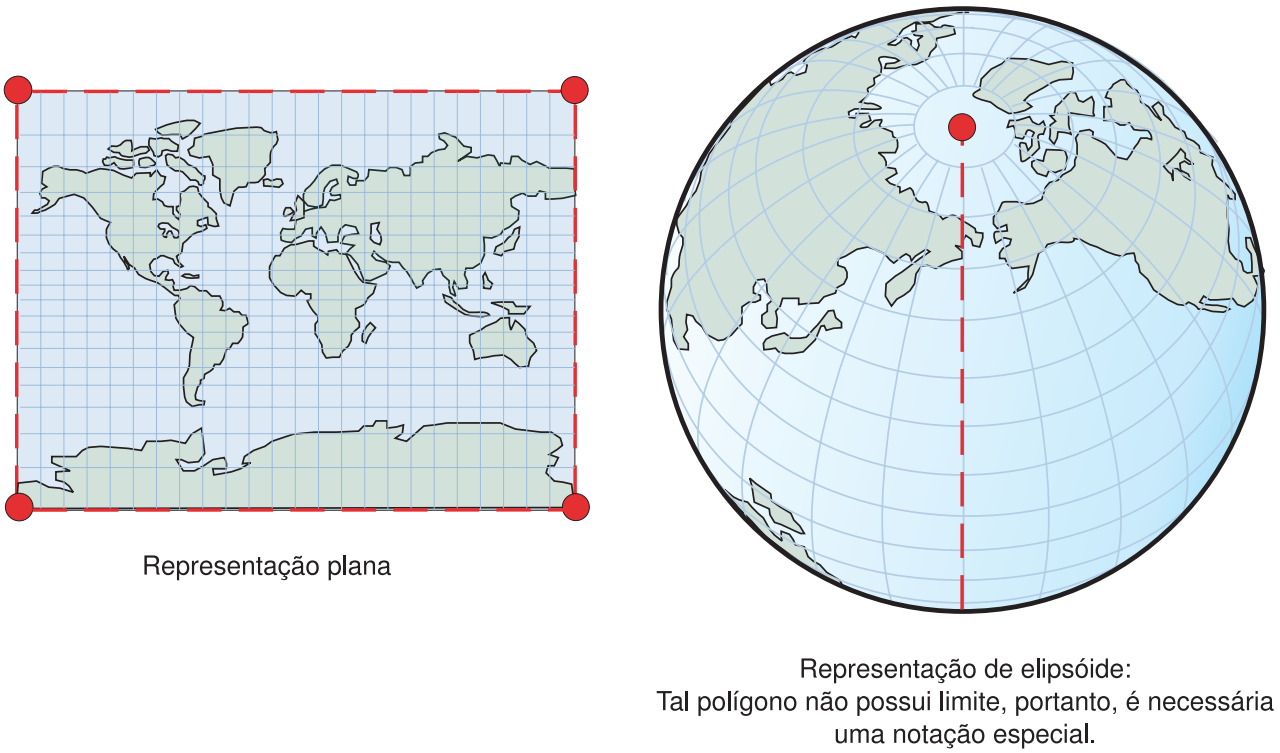


Figura 43. Polígonos que Representam Toda a Terra

Conceitos Relacionados:

- "Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender" na página 160
- "Regiões Geodésicas" na página 164
- "Latitude e Longitude Geodésicas" na página 161
- "Distâncias Geodésicas" na página 162
- "Esferóides Geodésicos" na página 222

Funções Espaciais Suportadas pelo DB2 Geodetic Extender

O DB2 Spatial Extender é baseado na biblioteca de funções fornecida pelo ESRI e o DB2 Geodetic Extender é baseado na biblioteca de funções Hipparchus. As diferenças entre a funcionalidade no ESRI e nas bibliotecas Hipparchus geram diferenças menores no comportamento de algumas funções. A tabela a seguir mostra as funções do Spatial Extender suportadas pelo Geodetic Extender e indica quaisquer diferenças no comportamento. Para obter informações sobre o uso e a sintaxe de funções espaciais, consulte o tópico de funções espaciais apropriado.

Tabela 26. Suporte de Funções para o Geodetic Extender

Função	Suportada pelo DB2 Geodetic Extender?	Diferença no Comportamento para o DB2 Geodetic Extender
EnvelopesIntersect	Sim	Nenhuma
MBR Aggregate	Não	Não aplicável
ST_AppendPoint	Não	Não aplicável
ST_Area	Sim	A unidade de medida padrão é metros.
ST_AsBinary	Sim	Nenhuma
ST_AsGML	Sim	Nenhuma
ST_AsShape	Sim	Nenhuma
ST_AsText	Sim	Nenhuma
ST_Boundary	Não	Não aplicável
ST_Buffer	Sim	Suportada apenas com pontos e multipontos. A distância pode ser um valor negativo. A unidade de medida padrão é metros.
ST_Centroid	Não	Não aplicável
ST_ChangePoint	Não	Não aplicável
ST_Contains	Sim	As duas geometrias devem estar no mesmo SRS (Spatial Reference System) geodésico.
ST_ConvexHull	Não	Não aplicável
ST_CoordDim	Sim	Nenhuma
ST_Crosses	Não	Não aplicável
ST_Difference	Sim	Não suportada com cadeias de linhas e cadeias multilinha. As duas geometrias devem estar no mesmo SRS geodésico. A dimensão da geometria retornada é igual à das geometrias de entrada.
ST_Dimension	Sim	Nenhuma
ST_Disjoint	Sim	Nenhuma
ST_Distance	Sim	Retorna a *distância geodésica*. As duas geometrias devem estar no mesmo SRS geodésico. A unidade de medida padrão é metros.
ST_Edge_GC_USA	Sim	Nenhuma
ST_Endpoint	Sim	Nenhuma
ST_Envelope	Sim	Envelope é um polígono que inclui o MBC (Minimum Bounding Circle) da geometria.
ST_EnvIntersects	Sim	Nenhuma
ST_EqualCoordsys	Sim	Nenhuma
ST_Equals	Não	Não aplicável
ST_EqualSRS	Sim	Nenhuma
ST_ExteriorRing	Sim	Nenhuma
ST_FindMeasure ou ST_LocateAlong	Não	Não aplicável
ST_Generalize	Sim	A unidade para o limite é metros.
ST_GeomCollection	Não	Não aplicável

Tabela 26. Suporte de Funções para o Geodetic Extender (continuação)

Função	Suportada pelo DB2 Geodetic Extender?	Diferença no Comportamento para o DB2 Geodetic Extender
ST_GeomCollFromTxt	Não	Não aplicável
ST_GeomCollFromWKB	Não	Não aplicável
ST_Geometry	Sim	Nenhuma
ST_GeometryN	Sim	Nenhuma
ST_GeometryType	Sim	Nenhuma
ST_GeomFromText	Sim	Nenhuma
ST_GeomFromWKB	Sim	Nenhuma
ST_GetIndexParms	Não	Não aplicável
ST_InteriorRingN	Sim	Nenhuma
ST_Intersection	Sim	A dimensão da geometria retornada é a dimensão da entrada com a dimensão inferior, exceto a dimensão da interseção de duas cadeias de linhas que é 0.
ST_Intersects	Sim	As duas geometrias devem estar no mesmo SRS geodésico.
ST_Is3d	Sim	Nenhuma
ST_IsClosed	Sim	Nenhuma
ST_IsEmpty	Sim	Nenhuma
ST_IsMeasured	Sim	Nenhuma
ST_IsRing	Não	Não aplicável
ST_IsSimple	Não	Não aplicável
ST_IsValid	Sim	Nenhuma
ST_Length	Sim	A unidade de medida padrão é metros.
ST_LineFromText	Sim	Nenhuma
ST_LineFromWKB	Sim	Nenhuma
ST_LineString	Sim	Nenhuma
ST_LineStringN	Sim	Nenhuma
ST_M	Sim	Nenhuma
ST_MaxM	Sim	Nenhuma
ST_MaxX	Sim	Retorna o valor máximo de X do MBC (Minimum Bounding Circle). **Nota:** Se o MBC cruzar o meridiano de data, o valor ST_MaxX será menor que ST_MinX. Se o MBC incluir o Pólo Norte, ou o Pólo Sul, ST_MinX será –180 e ST_MaxX será 180.
ST_MaxY	Sim	Retorna o valor máximo de Y do MBC. **Nota:** Se o MBC incluir o Pólo Norte, o valor ST_MaxY será 90.
ST_MaxZ	Sim	Nenhuma
ST_MBR	Sim	MBR é uma geometria que inclui o MBC da geometria.
ST_MBRIntersects	Sim	Nenhuma
ST_MeasureBetween ou ST_LocateBetween	Não	Não aplicável

Tabela 26. Suporte de Funções para o Geodetic Extender (continuação)

Função	Suportada pelo DB2 Geodetic Extender?	Diferença no Comportamento para o DB2 Geodetic Extender
ST_MidPoint	Sim	Nenhuma
ST_MinM	Sim	Nenhuma
ST_MinX	Sim	Retorna o valor mínimo de X do MBC. **Nota:** Se o MBC cruzar o meridiano de data, o valor ST_MinX será maior que o valor ST_MaxX. Se o MBC incluir o Pólo Norte, ou o Pólo Sul, ST_MinX será –180 e ST_MaxX será 180.
ST_MinY	Sim	Retorna o valor mínimo de Y do MBC. **Nota:** Se o MBC incluir o Pólo Sul, o valor ST_MinY será –90.
ST_MinZ	Sim	Nenhuma
ST_MLineFromText	Sim	Nenhuma
ST_MLineFromWKB	Sim	Nenhuma
ST_MPointFromText	Sim	Nenhuma
ST_MPointFromWKB	Sim	Nenhuma
ST_MPolyFromText	Sim	Nenhuma
ST_MPolyFromWKB	Sim	Nenhuma
ST_MultiLineString	Sim	Nenhuma
ST_MultiPoint	Sim	Nenhuma
ST_MultiPolygon	Sim	Nenhuma
ST_NumGeometries	Sim	Nenhuma
ST_NumInteriorRing	Sim	Nenhuma
ST_NumLineStrings	Sim	Nenhuma
ST_NumPoints	Sim	Nenhuma
ST_NumPolygons	Sim	Nenhuma
ST_Overlaps	Não	Não aplicável
ST_Perimeter	Sim	A unidade de medida padrão é metros.
ST_PerpPoints	Não	Não aplicável
ST_Point	Sim	Nenhuma
ST_PointFromText	Sim	Nenhuma
ST_PointFromWKB	Sim	Nenhuma
ST_PointN	Sim	Nenhuma
ST_PolyFromText	Sim	Nenhuma
ST_PolyFromWKB	Sim	Nenhuma
ST_PointOnSurface	Sim	Nenhuma
ST_Polygon	Sim	Nenhuma
ST_PolygonN	Sim	Nenhuma
ST_Relate	Não	Não aplicável
ST_RemovePoint	Não	Não aplicável
ST_SrsId ou ST_SRID	Sim	Nenhuma

Tabela 26. Suporte de Funções para o Geodetic Extender (continuação)

Função	Suportada pelo DB2 Geodetic Extender?	Diferença no Comportamento para o DB2 Geodetic Extender
ST_SrsName	Sim	Nenhuma
ST_StartPoint	Sim	Nenhuma
ST_SymDifference	Sim	Não suportada com cadeias de linhas e cadeias multilinha. A dimensão da geometria retornada é igual à das geometrias de entrada. As duas geometrias devem estar no mesmo SRS geodésico.
ST_ToGeomColl	Não	Não aplicável
ST_ToLineString	Sim	Nenhuma
ST_ToMultiLine	Sim	Nenhuma
ST_ToMultiPoint	Sim	Nenhuma
ST_ToPoint	Sim	Nenhuma
ST_ToPolygon	Sim	Nenhuma
ST_Touches	Não	Não aplicável
ST_Transform	Sim	Nenhuma. **Nota:** As transformações de coordenadas são feitas ponto por ponto. Quando fizer a transformação entre sistemas de coordenadas geodésicos e sistemas de coordenadas planares não projetados, verifique com atenção os polígonos e cadeias de linhas que estendem o meridiano de 180 graus ou incluem um ou os dois pólos. Como o Spatial Extender e o Geodetic Extender tratam estes casos de forma diferente, é possível que geometrias que são válidas em um sistema de coordenadas planas da Terra não sejam válidas em um sistema de Terra redonda e vice-versa. Para obter informações adicionais, consulte "Diferenças em Trabalhar com Representações Planas e Redondas da Terra" na página 199.
ST_Union	Sim	As duas geometrias devem estar no mesmo SRS geodésico.
ST_Within	Sim	As duas geometrias devem estar no mesmo SRS geodésico.
ST_WKBToSQL	Sim	Nenhuma
ST_WKTToSQL	Sim	Nenhuma
ST_X	Sim	Nenhuma
ST_Y	Sim	Nenhuma
ST_Z	Sim	Nenhuma
Union Aggregate	Não	Não aplicável

Conceitos Relacionados:

- "Quando Utilizar o DB2 Geodetic Extender e Quando Utilizar o DB2 Spatial Extender" na página 160
- "Regiões Geodésicas" na página 164
- "Latitude e Longitude Geodésicas" na página 161

- "Distâncias Geodésicas" na página 162

Tarefas Relacionadas:

- "Criando Índices Geodésicos de Voronoi" na página 181

Referência Relacionada:

- "Diferenças em Trabalhar com Representações Planas e Redondas da Terra" na página 199

Procedimentos Armazenados e Exibições de Catálogos do DB2 Geodetic Extender

O DB2 Geodetic Extender suporta as mesmas exibições de catálogos que o DB2 Spatial Extender e suporta um subconjunto dos procedimentos armazenados espaciais.

O Geodetic Extender não suporta os seguintes procedimentos armazenados:

- ST_disable_autogeocoding
- ST_enable_autogeocoding
- ST_register_geocoder
- ST_remove_geocoding_setup
- ST_run_geocoding
- ST_setup_geocoding
- ST_unregister_geocoder

O Geodetic Extender fornece 318 sistemas de referência espacial geodésicos predefinidos que aparecem na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS. Consulte "Datums Suportados pelo DB2 Geodetic Extender" para obter uma lista completa.

Datums Suportados pelo DB2 Geodetic Extender

Conforme o "Sistema de Coordenadas Geográficas" na página 59 descreve, um *datum* é um conjunto de valores que define a posição de um elipsóide com relação ao centro da Terra. Um sistema de referência espacial (SRS) é um conjunto de parâmetros que associam um datum a um elipsóide e é identificado com um SRID (Spatial Reference System Identifier). A Tabela 28 lista os datums predefinidos fornecidos pelo DB2 Geodetic Extender. Os valores de deslocamento e fatores de escala para todos os SRSs geodésicos predefinidos são os mesmos e a tabela a seguir mostra seus valores.

Tabela 27. Valores de Deslocamento e de Escala para SRSs Geodésicos Predefinidos

Parâmetro do SRS	Valor
xOffset	–180
yOffset	–90
zOffset	–50000
mOffset	–1000
xScale	5965232
yScale	5965232
zScale	1000

Tabela 27. Valores de Deslocamento e de Escala para SRSs Geodésicos Predefinidos (continuação)

Parâmetro do SRS	Valor
mScale	1000

O yScale é sempre igual ao xScale.

Você pode escolher qualquer datum listado na Tabela 28 para seu sistema de referência espacial. É recomendável escolher um que seja mais adequado a seus dados. Por exemplo, um dos datums mais comumente utilizado, o World Geodetic System 1984 (WGS 1984), utiliza o centro da Terra como seu ponto de origem e mapeia todo o globo; é um datum centralizado na Terra. Em contraste, um datum regional, como o datum North American 1927, mapeia a América do Norte começando de um ponto no solo. Um datum regional é exato para a região e tem como objeto a modelagem, mas um datum geodésico centralizado na Terra é necessário para tratar localizações em todo o globo.

Tabela 28. SRIDs com Datum e Elipsóide Associados

SRID	Nome do Datum	Elipsóide de Referência
2000000000	WGS 1984	WGS 1984
2000000001	Abidjan 1987	Clarke 1880 (RGS)
2000000002	Accra	War Office
2000000003	Adindan	Clarke 1880 (RGS)
2000000004	Afgooye	Krasovsky 1940
2000000005	Agadez	Clarke 1880 (IGN)
2000000006	Dados geodésicos australianos 1966	Australiano
2000000007	Dados geodésicos australianos 1984	Australiano
2000000008	Ain el Abd 1970	Internacional 1924
2000000009	Airy 1830	Airy 1830
2000000010	Airy Modified	Airy Modified
2000000011	Alaskan Islands	Clarke 1866
2000000012	Amersfoort	Bessel 1841
2000000013	Anguilla 1957	Clarke 1880 (RGS)
2000000014	Anna 1 Astro 1965	Australiano
2000000015	Antigua Astro 1943	Clarke 1880 (RGS)
2000000016	Aratu	Internacional 1924
2000000017	Arc 1950	Clarke 1880 (Arc)
2000000018	Arc 1960	Clarke 1880 (RGS)
2000000019	Ascension Island 1958	Internacional 1924
2000000020	Assumed Geographic (NAD27 para arquivos modelo sem um PRJ)	Clarke 1866
2000000021	Astronomical Station 1952	Internacional 1924
2000000022	ATF (Paris)	Plessis 1817
2000000023	Average Terrestrial System 1977	ATS 1977
2000000024	Australian National	Australiano

Tabela 28. SRIDs com Datum e Elipsóide Associados (continuação)

SRID	Nome do Datum	Elipsóide de Referência
2000000025	Ayabelle Lighthouse	Clarke 1880 (RGS)
2000000026	Bab South Astro (Bablethuap Is, Republic of Palau)	Clarke 1866
2000000027	Barbados 1938	Clarke 1880 (RGS)
2000000028	Batavia	Bessel 1841
2000000029	Batavia (Jakarta)	Bessel 1841
2000000030	Astro Beacon E 1945	Internacional 1924
2000000031	Beduaram	Clarke 1880 (IGN)
2000000032	Beijing 1954	Krasovsky 1940
2000000033	Reseau National Belge 1950	Internacional 1924
2000000034	Belge 1950 (Brussels)	Internacional 1924
2000000035	Reseau National Belge 1972	Internacional 1924
2000000036	Bellevue (IGN)	Internacional 1924
2000000037	Bermuda 1957	Clarke 1866
2000000038	Bern 1898	Bessel 1841
2000000039	Bern 1898 (Bern)	Bessel 1841
2000000040	Bern 1938	Bessel 1841
2000000041	Bessel 1841	Bessel 1841
2000000042	Bessel Modificado	Bessel Modificado
2000000043	Bessel Namibia	Bessel Namibia
2000000044	Bissau	Internacional 1924
2000000045	Bogota	Internacional 1924
2000000046	Bogota (Bogota)	Internacional 1924
2000000047	Bukit Rimpah	Bessel 1841
2000000048	Camacupa	Clarke 1880 (RGS)
2000000049	Campo Inchauspe	Internacional 1924
2000000050	Camp Area Astro	Internacional 1924
2000000051	Canton Astro 1966	Internacional 1924
2000000052	Cape	Clarke 1880 (Arc)
2000000053	Cape Canaveral	Clarke 1866
2000000054	Carthage	Clarke 1880 (IGN)
2000000055	Carthage (graus)	Clarke 1880 (IGN)
2000000056	Carthage (Paris)	Clarke 1880 (IGN)
2000000057	CH 1903	Bessel 1841
2000000058	CH 1903+	Bessel 1841
2000000059	Chatham Island Astro 1971	Internacional 1924
2000000060	Chos Malal 1914	Internacional 1924
2000000061	Swiss Terrestrial Ref. Frame 1995	GRS 1980
2000000062	Chua	Internacional 1924

Tabela 28. SRIDs com Datum e Elipsóide Associados (continuação)

SRID	Nome do Datum	Elipsóide de Referência
2000000063	Clarke 1858	Clarke 1858
2000000064	Clarke 1866	Clarke 1866
2000000065	Clarke 1866 (Michigan)	Clarke 1866 (Michigan)
2000000066	Clarke 1880	Clarke 1880
2000000067	Clarke 1880 (Arc)	Clarke 1880 (Arc)
2000000068	Clarke 1880 (Benoit)	Clarke 1880 (Benoit)
2000000069	Clarke 1880 (IGN)	Clarke 1880 (IGN)
2000000070	Clarke 1880 (RGS)	Clarke 1880 (RGS)
2000000071	Clarke 1880 (SGA)	Clarke 1880 (SGA)
2000000072	Conakry 1905	Clarke 1880 (IGN)
2000000073	Corrego Alegre	Internacional 1924
2000000074	Cote d'Ivoire	Clarke 1880 (IGN)
2000000075	Dabola 1981	Clarke 1880 (RGS)
2000000076	Datum 73	Internacional 1924
2000000077	Dealul Piscului 1933 (Romania)	Internacional 1924
2000000078	Dealul Piscului 1970 (Romania)	Krasovsky 1940
2000000079	Deception Island	Clarke 1880 (RGS)
2000000080	Deir ez Zor	Clarke 1880 (IGN)
2000000081	Deutsche Hauptdreiecksnetz	Bessel 1841
2000000082	Dominica 1945	Clarke 1880 (RGS)
2000000083	DOS 1968	Internacional 1924
2000000084	Astro DOS 71/4	Internacional 1924
2000000085	Douala	Clarke 1880 (IGN)
2000000086	Easter Island 1967	Internacional 1924
2000000087	European Datum 1950	Internacional 1924
2000000088	European Datum 1950 (ED77)	Internacional 1924
2000000089	European Datum 1987	Internacional 1924
2000000090	Egito 1907	Helmert 1906
2000000091	Estonia 1937	Bessel 1841
2000000092	Estonia 1992	GRS 1980
2000000093	European Terrestrial Ref. Frame 1989	WGS 1984
2000000094	European 1979	Internacional 1924
2000000095	European Libyan Datum 1979	Internacional 1924
2000000096	Everest 1830	Everest 1830
2000000097	Everest (Bangladesh)	Everest Adjustment 1937
2000000098	Everest (Definition 1962)	Everest (Definition 1962)

Tabela 28. SRIDs com Datum e Elipsóide Associados (continuação)

SRID	Nome do Datum	Elipsóide de Referência
2000000099	Everest (Definition 1967)	Everest (Definition 1967)
2000000100	Everest (Definition 1975)	Everest (Definition 1975)
2000000101	Everest (India and Nepal)	Everest (Definition 1962)
2000000102	Everest 1830 Modified	Everest 1830 Modified
2000000103	Everest Modified 1969	Everest Modified 1969
2000000104	Fahud	Clarke 1880 (RGS)
2000000105	Final Datum 1958	Clarke 1880 (RGS)
2000000106	Fischer 1960	Fischer 1960
2000000107	Fischer 1968	Fischer 1968
2000000108	Fischer Modified	Fischer Modified
2000000109	Fort Thomas 1955	Clarke 1880 (RGS)
2000000110	Gandajika 1970	Internacional 1924
2000000111	Gan 1970	Internacional 1924
2000000112	Garoua	Clarke 1880 (IGN)
2000000113	Geocentric Datum of Australia 1994	GRS 1980
2000000114	GEM 10C Gravity Potential Model	GEM 10C
2000000115	Greek Geodetic Ref. System 1987	GRS 1980
2000000116	Graciosa Base SW 1948	Internacional 1924
2000000117	Grego	Bessel 1841
2000000118	Greek (Athens)	Bessel 1841
2000000119	Grenada 1953	Clarke 1880 (RGS)
2000000120	GRS 1967	GRS 1967
2000000121	GRS 1980	GRS 1980
2000000122	Guam 1963	Clarke 1866
2000000123	Gunung Segara	Bessel 1841
2000000124	GUX 1 Astro	Internacional 1924
2000000125	Guiana Francesa	Internacional 1924
2000000126	Hanoi 1972	Krasovsky 1940
2000000127	Hartebeesthoek 1994	WGS 1984
2000000128	Helmert 1906	Helmert 1906
2000000129	Herat North	Internacional 1924
2000000130	Hermannskogel	Bessel 1841
2000000131	Hito XVIII 1963	Internacional 1924
2000000132	Hjorsey 1955	Internacional 1924
2000000133	Hong Kong 1963	Internacional 1924
2000000134	Hong Kong 1980	Internacional 1924
2000000135	Hough 1960	Hough 1960

Tabela 28. SRIDs com Datum e Elipsóide Associados (continuação)

SRID	Nome do Datum	Elipsóide de Referência
2000000136	Hungarian Datum 1972	GRS 1967
2000000137	Hu Tzu Shan	Internacional 1924
2000000138	Indian 1954	Everest Adjustment 1937
2000000139	Indian 1960	Everest Adjustment 1937
2000000140	Indian 1975	Everest Adjustment 1937
2000000141	Indonesian National	Indonesian National
2000000142	Indonesian Datum 1974	Indonesian
2000000143	International 1927	Internacional 1924
2000000144	International 1967	International 1967
2000000145	IRENET95	GRS 1980
2000000146	Israel	GRS 1980
2000000147	ISTS 061 Astro 1968	Internacional 1924
2000000148	ISTS 073 Astro 1969	Internacional 1924
2000000149	Jamaica 1875	Clarke 1880
2000000150	Jamaica 1969	Clarke 1866
2000000151	Japan Geodetic Datum 2000	GRS 1980
2000000152	Johnston Island 1961	Internacional 1924
2000000153	Kalianpur 1880	Everest 1830
2000000154	Kalianpur 1937	Everest Adjustment 1937
2000000155	Kalianpur 1962	Everest (Definition 1962)
2000000156	Kalianpur 1975	Everest (Definition 1975)
2000000157	Kandawala	Everest Adjustment 1937
2000000158	Kerguelen Island 1949	Internacional 1924
2000000159	Kertau	Everest 1830 Modified
2000000160	Kartastokoordinaattijarjestelma	Internacional 1924
2000000161	Kuwait Oil Company	Clarke 1880 (RGS)
2000000162	Korean Datum 1985	Bessel 1841
2000000163	Korean Datum 1995	WGS 1984
2000000164	Krasovsky 1940	Krasovsky 1940
2000000165	Kuwait Utility	GRS 1980
2000000166	Kusaie Astro 1951	Internacional 1924
2000000167	Lake	Internacional 1924
2000000168	La Canoa	Internacional 1924
2000000169	L.C. 5 Astro 1961	Clarke 1866

Tabela 28. SRIDs com Datum e Elipsóide Associados (continuação)

SRID	Nome do Datum	Elipsóide de Referência
2000000170	Leigon	Clarke 1880 (RGS)
2000000171	Libéria 1964	Clarke 1880 (RGS)
2000000172	Datum Lisboa Bessel	Bessel 1841
2000000173	Datum Lisboa Hayford	Internacional 1924
2000000174	Lisboa	Internacional 1924
2000000175	Lisbon (Lisbon)	Internacional 1924
2000000176	LKS 1994	GRS 1980
2000000177	Locodjo 1965	Clarke 1880 (RGS)
2000000178	Loma Quintana	Internacional 1924
2000000179	Lome	Clarke 1880 (IGN)
2000000180	Luzon 1911	Clarke 1866
2000000181	Madrid 1870 (Madrid Prime Merid.)	Struve 1860
2000000182	Madzansua	Clarke 1866
2000000183	Mahe 1971	Clarke 1880 (RGS)
2000000184	Majuro (Republic of Marshall Is.)	Clarke 1866
2000000185	Makassar	Bessel 1841
2000000186	Makassar (Jakarta)	Bessel 1841
2000000187	Malongo 1987	Internacional 1924
2000000188	Manoca	Clarke 1880 (RGS)
2000000189	Massawa	Bessel 1841
2000000190	Merchich	Clarke 1880 (IGN)
2000000191	Merchich (graus)	Clarke 1880 (IGN)
2000000192	Militar-Geographische Institut	Bessel 1841
2000000193	MGI (Ferro)	Bessel 1841
2000000194	Mhast	Internacional 1924
2000000195	Midway Astro 1961	Internacional 1924
2000000196	Minna	Clarke 1880 (RGS)
2000000197	Monte Mario	Internacional 1924
2000000198	Monte Mario (Rome)	Internacional 1924
2000000199	Montserrat Astro 1958	Clarke 1880 (RGS)
2000000200	Mount Dillon	Clarke 1858
2000000201	Moznet	WGS 1984
2000000202	M'poraloko	Clarke 1880 (IGN)
2000000203	North American Datum 1927	Clarke 1866
2000000204	NAD 1927 CGQ77	Clarke 1866
2000000205	NAD 1927 (1976)	Clarke 1866
2000000206	North American Datum 1983	GRS 1980
2000000207	NAD 1983 (Canadian Spatial Ref. System)	GRS 1980
2000000208	North American Datum 1983 (HARN)	GRS 1980

Tabela 28. SRIDs com Datum e Elipsóide Associados (continuação)

SRID	Nome do Datum	Elipsóide de Referência
2000000209	NAD Michigan	Clarke 1866 (Michigan)
2000000210	Nahrwan 1967	Clarke 1880 (RGS)
2000000211	Naparima 1955	Internacional 1924
2000000212	Naparima 1972	Internacional 1924
2000000213	Nord de Guerre (Paris)	Plessis 1817
2000000214	National Geodetic Network (Kuwait)	WGS 1984
2000000215	NGO 1948	Bessel Modificado
2000000216	NGO 1948 (Oslo)	Bessel Modificado
2000000217	Nord Sahara 1959	Clarke 1880 (RGS)
2000000218	NSWC 9Z-2	NWL 9D
2000000219	Nouvelle Triangulation Francaise (graus)	Clarke 1880 (IGN)
2000000220	NTF (Paris) (grads)	Clarke 1880 (IGN)
2000000221	NWL 9D Transit Precise Ephemeris	NWL 9D
2000000222	New Zealand Geodetic Datum 1949	Internacional 1924
2000000223	New Zealand Geodetic Datum 2000	GRS 1980
2000000224	Observatario	Clarke 1866
2000000225	Observ. Meteorologico 1939	Internacional 1924
2000000226	Old Hawaiian	Clarke 1866
2000000227	Oman	Clarke 1880 (RGS)
2000000228	OSGB 1936	Airy 1830
2000000229	OSGB 1970 (SN)	Airy 1830
2000000230	OSU 1986 Geoidal Model	OSU 86F
2000000231	OSU 1991 Geoidal Model	OSU 91A
2000000232	OS (SN) 1980	Airy 1830
2000000233	Padang 1884	Bessel 1841
2000000234	Padang 1884 (Jakarta)	Bessel 1841
2000000235	Palestina 1923	Clarke 1880 (Benoit)
2000000236	Pampa del Castillo	Internacional 1924
2000000237	PDO Survey Datum 1993	Clarke 1880 (RGS)
2000000238	Pico de Las Nieves	Internacional 1924
2000000239	Pitcairn Astro 1967	Internacional 1924
2000000240	Plessis 1817	Plessis 1817
2000000241	Pohnpei (Fed. States of Micronesia)	Clarke 1866
2000000242	Point 58	Clarke 1880 (RGS)
2000000243	Pointe Noire	Clarke 1880 (IGN)
2000000244	Porto Santo 1936	Internacional 1924
2000000245	POSGAR	GRS 1980
2000000246	Provisional South Amer. Datum 1956	Internacional 1924

Tabela 28. SRIDs com Datum e Elipsóide Associados (continuação)

SRID	Nome do Datum	Elipsóide de Referência
2000000247	Puerto Rico	Clarke 1866
2000000248	Pulkovo 1942	Krasovsky 1940
2000000249	Pulkovo 1995	Krasovsky 1940
2000000250	Qatar 1974	Internacional 1924
2000000251	Qatar 1948	Helmert 1906
2000000252	Qornoq	Internacional 1924
2000000253	Rassadiran	Internacional 1924
2000000254	REGVEN	GRS 1980
2000000255	Reunion	Internacional 1924
2000000256	Reseau Geodesique Francais 1993	GRS 1980
2000000257	RT38	Bessel 1841
2000000258	RT38 (Stockholm)	Bessel 1841
2000000259	RT 1990	Bessel 1841
2000000260	S-42 Hungary	Krasovsky 1940
2000000261	South American Datum 1969	GRS 1967 Truncado
2000000262	Samboja	Bessel 1841
2000000263	American Samoa 1962	Clarke 1866
2000000264	Santo DOS 1965	Internacional 1924
2000000265	Sao Braz	Internacional 1924
2000000266	Sapper Hill 1943	Internacional 1924
2000000267	Schwarzeck	Bessel Namibia
2000000268	Segora	Bessel 1841
2000000269	Selvagem Grande 1938	Internacional 1924
2000000270	Serindung	Bessel 1841
2000000271	Sierra Leone 1924	War Office
2000000272	Sierra Leone 1960	Clarke 1880 (RGS)
2000000273	Sierra Leone 1968	Clarke 1880 (RGS)
2000000274	SIRGAS	GRS 1980
2000000275	South Yemen	Krasovsky 1940
2000000276	Authalic sphere	Sphere
2000000277	Authalic sphere (ARC/INFO)	Sphere ARC INFO
2000000278	Struve 1860	Struve 1860
2000000279	St. George Island (Alaska)	Clarke 1866
2000000280	St. Kitts 1955	Clarke 1880 (RGS)
2000000281	St. Lawrence Island (Alaska)	Clarke 1866
2000000282	St. Lucia 1955	Clarke 1880 (RGS)
2000000283	St. Paul Island (Alaska)	Clarke 1866
2000000284	St. Vincent 1945	Clarke 1880 (RGS)
2000000285	Sudan	Clarke 1880 (IGN)

Tabela 28. SRIDs com Datum e Elipsóide Associados (continuação)

SRID	Nome do Datum	Elipsóide de Referência
2000000286	South Asia Singapore	Fischer Modified
2000000287	S-JTSK	Bessel 1841
2000000288	S-JTSK (Ferro)	Bessel 1841
2000000289	Tananarive 1925	Internacional 1924
2000000290	Tananarive 1925 (Paris)	Internacional 1924
2000000291	Tern Island Astro 1961	Internacional 1924
2000000292	Tete	Clarke 1866
2000000293	Timbalai 1948	Everest (Definition 1967)
2000000294	TM65	Airy Modified
2000000295	TM75	Airy Modified
2000000296	Tokyo	Bessel 1841
2000000297	Trinidad 1903	Clarke 1858
2000000298	Tristan Astro 1968	Internacional 1924
2000000299	Trucial Coast 1948	Helmert 1906
2000000300	Viti Levu 1916	Clarke 1880 (RGS)
2000000301	Voirol 1875	Clarke 1880 (IGN)
2000000302	Voirol 1875 (degrees)	Clarke 1880 (IGN)
2000000303	Voirol 1875 (Paris)	Clarke 1880 (IGN)
2000000304	Voirol Unifie 1960	Clarke 1880 (RGS)
2000000305	Voirol Unifie 1960 (graus)	Clarke 1880 (RGS)
2000000306	Voirol Unifie 1960 (Paris)	Clarke 1880 (RGS)
2000000307	Wake-Eniwetok 1960	Hough 1960
2000000308	Wake Island Astro 1952	Internacional 1924
2000000309	Walbeck	Walbeck
2000000310	War Office	War Office
2000000311	WGS 1966	WGS 1966
2000000312	WGS 1972	WGS 1972
2000000313	WGS 1972 Transit Broadcast Ephemeris	WGS 1972
2000000314	Yacare	Internacional 1924
2000000315	Yemen Nat'l Geodetic Network 1996	WGS 1984
2000000316	Yoff	Clarke 1880 (IGN)
2000000317	Zanderij	Internacional 1924

Esferóides Geodésicos

Um esferóide (também conhecido como um elipsóide) faz parte de um sistema de coordenadas geográficas que define o formato da superfície da Terra em uma localização específica.

A definição de um sistema de coordenadas inclui a definição de um elipsóide na definição SPHEROID que faz parte da definição DATUM, conforme mostra o exemplo a seguir:

```
GEOGCS["GCS_North_American_1983",DATUM["D_North_American_1983",
SPHEROID["GRS_1980",6378137,298.257222101]],
PRIMEM["Greenwich",0],UNIT["Degree",0.0174532925199432955]]
```

Para obter uma lista dos esferóides fornecidos pelo Spatial Extender e pelo Geodetic Extender, consulte "Sistemas de Coordenadas Suportados" na página 529. Você pode utilizar a exibição do catálogo DB2GSE.ST_COORDINATE_SYSTEMS para recuperar estas informações. A coluna **DEFINITION** na exibição do catálogo DB2GSE.ST_COORDINATE_SYSTEMS contém os valores nas colunas **Nome**, **Semi-eixo maior** e **Achatado** na tabela Esferóides Suportados.

Conceitos Relacionados:

- "Sistema de Coordenadas Geográficas" na página 59

Referência Relacionada:

- "A Exibição do Catálogo DB2GSE.ST_COORDINATE_SYSTEMS" na página 287
- "ST_create_coordsys" na página 238

Parte 5. Material de Referência

Capítulo 20. Procedimentos Armazenados

Esta seção fornece informações de referência sobre os procedimentos armazenados que você pode utilizar para configurar o DB2 Spatial Extender e criar projetos que utilizam dados espaciais. Ao configurar o DB2 Spatial Extender ou criar projetos a partir do Centro de Controle do DB2 ou do processador de linha de comandos do DB2, você chama estes procedimentos armazenados implicitamente. Por exemplo, quando você clica em **OK** a partir de uma janela do DB2 Spatial Extender no Centro de Controle do DB2, o DB2 chama o procedimento armazenado que está associado a essa janela.

Como alternativa, você pode chamar os procedimentos armazenados num programa da aplicação.

Antes de chamar a maioria dos procedimentos armazenados do DB2 Spatial Extender em um banco de dados, você deve ativar esse banco de dados para operações espaciais chamando o procedimento armazenado ST_enable_db, diretamente ou utilizando o Centro de Controle do DB2. (Você pode ler sobre como chamar este procedimento armazenado no tópico sobre ST_enable_db, posteriormente nesta seção.)

Depois que um banco de dados é ativado para operações espaciais, você pode chamar qualquer procedimento armazenado do DB2 Spatial Extender, implicitamente ou explicitamente, nesse banco de dados, se estiver conectado ao mesmo.

Este capítulo fornece tópicos para todos os procedimentos armazenados do DB2 Spatial Extender, conforme a seguir:

- GSE_export_sde
- GSE_import_sde
- ST_alter_coordsys
- ST_alter_srs
- ST_create_coordsys
- ST_create_srs
- ST_disable_autogeocoding
- ST_disable_db
- ST_drop_coordsys
- ST_drop_srs
- ST_enable_autogeocoding
- ST_enable_db
- ST_export_shape
- ST_import_shape
- ST_register_geocoder
- ST_register_spatial_column
- ST_remove_geocoding_setup
- ST_run_geocoding
- ST_setup_geocoding
- ST_unregister_geocoder

- ST_unregister_spatial_column

As implementações dos procedimentos armazenados estão arquivadas na biblioteca db2gse do servidor do DB2 Spatial Extender.

GSE_export_sde

Utilize este procedimento armazenado para exportar uma coluna espacial e sua tabela associada para um arquivo de transferência SDE.

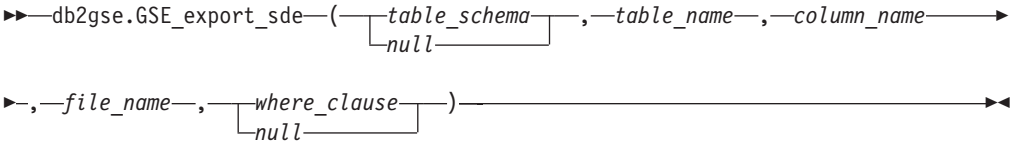
Restrições:

- Deve existir exatamente uma coluna espacial na tabela ou exibição.
- A coluna espacial deve estar registrada.
- Você não pode anexar a arquivos SDE existentes.

Autorização:

O ID de usuário sob o qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM. Além disso, este ID de usuário deve ter o privilégio SELECT sobre a tabela a ser exportada.

Sintaxe:



Descrições de parâmetros:

table_schema
Nomeia o esquema ao qual pertence a tabela especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela que está sendo exportada. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor *table_name* é convertido em maiúsculas, a menos que você o coloque entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Nomeia a coluna espacial registrada que está sendo exportada. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

file_name
Nomeia o arquivo de transferência SDE para o qual a coluna espacial especificada e sua tabela associada serão exportadas. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é VARCHAR(256).

where_clause
Especifica o conteúdo da cláusula SQL WHERE, que define uma restrição do conjunto de registros a ser exportado. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma restrição será definida na cláusula WHERE.

Se este parâmetro for especificado, o valor poderá fazer referência a qualquer coluna de atributo na tabela que você está exportando.

O tipo de dados deste parâmetro é VARCHAR(1024).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado GSE_export_sde. Este exemplo utiliza um comando DB2 CALL para exportar dados de uma tabela denominada CLIENTES para arquivos SDE:

```
call db2gse.GSE_export_sde(NULL,'CLIENTES','LOCALIZAÇÃO',
     '/tmp/export_sde_file', NULL,?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- “GSE_import_sde” na página 230

GSE_import_sde

Utilize este procedimento armazenado para importar um arquivo de transferência SDE para um banco de dados que esteja ativado para operações espaciais. O procedimento armazenado pode operar de uma destas duas formas:

- Se o arquivo de transferência SDE destinar-se a uma tabela existente que tenha uma coluna espacial registrada, o DB2 Spatial Extender carregará a tabela com os dados do arquivo.
- Caso contrário, o DB2 Spatial Extender criará uma tabela que tem uma coluna espacial, registrará esta coluna e carregará a coluna espacial e as outras colunas da tabela com os dados do arquivo.

O sistema de referência espacial especificado no arquivo de transferência SDE é comparado com os sistemas de referência espacial que são registrados no DB2 Spatial Extender. Se o sistema especificado corresponder a um sistema registrado, todos os valores dos dados de transferência, quando carregados, serão modificados do modo estabelecido pelo sistema registrado. Se o sistema especificado não corresponder a nenhum dos sistemas registrados, o DB2 Spatial Extender criará um novo sistema de referência espacial para especificar as modificações.

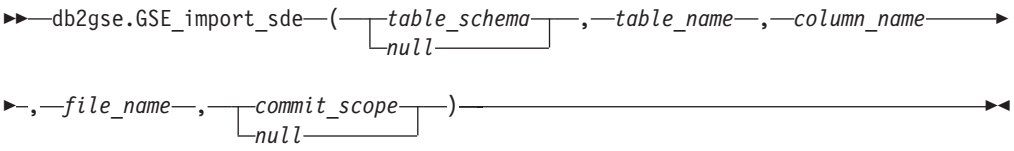
Autorização:

Ao importar dados para uma tabela existente, a ID de usuário com a qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela para a qual os dados serão importados
- Privilégio CONTROL nesta tabela

Quando a tabela para a qual você deseja importar dados deve ser criada, o ID de usuário sob o qual este procedimento armazenado é chamado deve conter autoridade SYSADM ou DBADM sobre o banco de dados que contém a tabela que será criada.

Sintaxe:



Descrições de parâmetros:

table_schema
Nomeia o esquema ao qual pertence a tabela ou exibição especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela na qual os dados de transferência SDE serão carregados. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Nomeia a coluna registrada na qual os dados espaciais do arquivo de transferência SDE serão carregados. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

file_name
Nomeia o arquivo de transferência SDE a ser importado. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é VARCHAR(256).

commit_scope
Especifica o número de registros que serão importados antes de um COMMIT ser emitido. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 0 (zero) será utilizado e nenhum registro será consolidado.

O tipo de dados deste parâmetro é INTEGER.

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado GSE_import_sde. Este exemplo utiliza um comando DB2 CALL para importar um arquivo SDE denominado

tmp/customerSDE para uma tabela denominada CLIENTES. Este comando CALL especifica que um COMMIT será executado após cada 5 registros serem importados:

```
call db2gse.GSE_import_sde(NULL,'CLIENTES','LOCALIZAÇÃO',
     '/tmp/customerSde', 5, ?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "GSE_export_sde" na página 228

ST_alter_coordsys

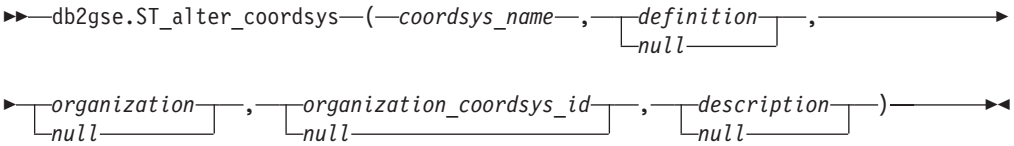
Utilize este procedimento armazenado para atualizar uma definição do sistema de coordenadas no banco de dados. Quando este procedimento armazenado é processado, as informações sobre o sistema de coordenadas são atualizadas na exibição do catálogo DB2GSE.ST_COORDINATE_SYSTEMS.

Atenção: Utilize este procedimento armazenado com atenção. Se você utilizar este procedimento para alterar a definição do sistema de coordenadas, e tiver dados espaciais existentes que estejam associados a um sistema de referência espacial que baseia-se nesse sistema de coordenadas, os dados espaciais poderão ser alterados inadvertidamente. Se os dados espaciais forem afetados, você será responsável por assegurar que os dados espaciais alterados ainda sejam precisos e válidos.

Autorização:

O ID de usuário sob o qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM.

Sintaxe:



Descrições de parâmetros:

coordsys_name
Identifica exclusivamente o sistema de coordenadas. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *coordsys_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

definition
Defina o sistema de coordenadas. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, a definição do sistema de coordenadas não será alterado.

O tipo de dados deste parâmetro é VARCHAR(2048).

organization
Nomeia a organização que definiu o sistema de coordenadas e forneceu a definição do mesmo; por exemplo, "European Petroleum Survey Group (EPSG)". Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo.

Se este parâmetro for nulo, a organização do sistema de coordenadas não foi alterado. Se este parâmetro não for nulo, o parâmetro *organization_coordsys_id* não poderá ser nulo; nesse caso, a combinação dos parâmetros *organization* e *organization_coordsys_id* identifica exclusivamente o sistema de coordenadas.

O tipo de dados deste parâmetro é VARCHAR(128).

organization_coordsys_id
Especifica um identificador numérico que está associado a este sistema de coordenadas pela entidade listada no parâmetro *organization*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo.

Se este parâmetro for nulo, o parâmetro *organization* também deverá ser nulo; nesse caso, o identificador do sistema de coordenadas da organização não é alterado. Se este parâmetro não for nulo, o parâmetro *organization* não poderá ser nulo; nesse caso, a combinação dos parâmetros *organization* e *organization_coordsys_id* identifica exclusivamente o sistema de coordenadas.

O tipo de dados deste parâmetro é INTEGER.

description
Descreve o sistema de coordenadas, explicando sua finalidade. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, as informações de descrição sobre o sistema de coordenadas não serão alteradas.

O tipo de dados deste parâmetro é VARCHAR(256).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_alter_coordsys. Este exemplo utiliza um comando DB2 CALL para atualizar um sistema de coordenadas denominado NORTH_AMERICAN_TEST. Este comando CALL atribui um valor de 1002 ao parâmetro *coordsys_id*:

```
call db2gse.ST_alter_coordsys('NORTH_AMERICAN_TEST',NULL,NULL,1002,NULL,?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_create_coordsys" na página 238
- "ST_drop_coordsys" na página 250

ST_alter_srs

Utilize este procedimento armazenado para atualizar uma definição do sistema de referência espacial no banco de dados. Quando este procedimento armazenado é processado, as informações sobre o sistema de referência espacial são atualizadas na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS.

Internamente, o DB2 Spatial Extender armazena os valores de coordenada como inteiros positivos. Portanto, durante o cálculo, o impacto de erros de arredondamento (que são seriamente dependentes do valor real para operações de ponto flutuante) pode ser reduzido. O desempenho das operações espaciais também pode melhorar significativamente.

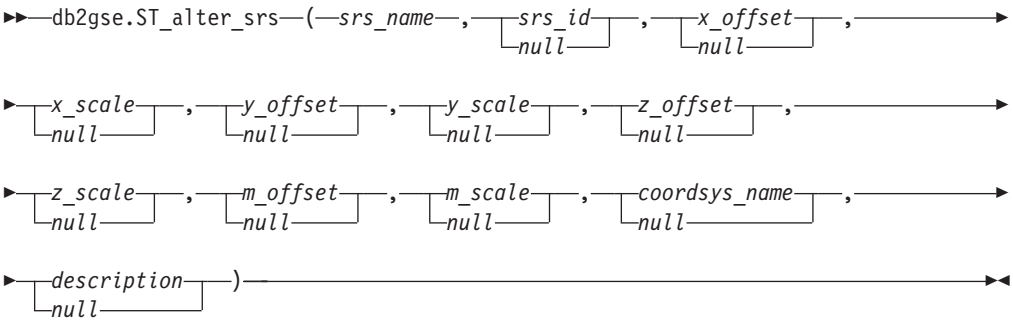
Restrição: Você não pode alterar um sistema de referência espacial se uma coluna espacial registrada o utiliza.

Atenção: Utilize este procedimento armazenado com atenção. Se você utilizar este procedimento para alterar o deslocamento, escala ou parâmetros *coordsys_name* do sistema de referência espacial, e tiver dados espaciais associados ao sistema de referência espacial, os dados espaciais poderão ser alterados inadvertidamente. Se os dados espaciais forem afetados, você será responsável por assegurar que os dados espaciais alterados ainda sejam precisos e válidos.

Autorização:

O ID de usuário sob o qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM.

Sintaxe:



Descrições de parâmetros:

srs_name
Identifica o sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *srs_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

srs_id
Identifica exclusivamente o sistema de referência espacial. Este identificador é utilizado como um parâmetro de entrada para várias funções espaciais. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o identificador numérico do sistema de referência espacial não será alterado.

O tipo de dados deste parâmetro é INTEGER.

x_offset
Especifica o deslocamento de todas as coordenadas X de geometrias que são representadas neste sistema de referência espacial. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor para este parâmetro na definição do sistema de referência espacial não será alterado.

O deslocamento é subtraído antes do fator de escala *x_scale* ser aplicado quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. (*WKT* é texto reconhecido e *WKB* é binário reconhecido.)

O tipo de dados deste parâmetro é DOUBLE.

x_scale
Especifica o fator de escala de todas as coordenadas X de geometrias que são representadas neste sistema de referência espacial. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor para este parâmetro na definição do sistema de referência espacial não será alterado.

O fator de escala é aplicado (multiplicação) após o deslocamento *x_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender.

O tipo de dados deste parâmetro é DOUBLE.

y_offset
Especifica o deslocamento de todas as coordenadas Y de geometrias que são representadas neste sistema de referência espacial. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor para este parâmetro na definição do sistema de referência espacial não será alterado.

O deslocamento é subtraído antes do fator de escala *y_scale* ser aplicado quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender.

O tipo de dados deste parâmetro é DOUBLE.

y_scale
Especifica o fator de escala de todas as coordenadas Y de geometrias que são representadas neste sistema de referência espacial. Embora você deva

especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor para este parâmetro na definição do sistema de referência espacial não será alterado.

O fator de escala é aplicado (multiplicação) após o deslocamento *y_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. Este fator de escala deve ser igual a *x_scale*.

O tipo de dados deste parâmetro é DOUBLE.

z_offset
Especifica o deslocamento de todas as coordenadas Z de geometrias que são representadas neste sistema de referência espacial. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor para este parâmetro na definição do sistema de referência espacial não será alterado.

O deslocamento é subtraído antes do fator de escala *z_scale* ser aplicado quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender.

O tipo de dados deste parâmetro é DOUBLE.

z_scale
Especifica o fator de escala de todas as coordenadas Z de geometrias que são representadas neste sistema de referência espacial. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor para este parâmetro na definição do sistema de referência espacial não será alterado.

O fator de escala é aplicado (multiplicação) após o deslocamento *z_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender.

O tipo de dados deste parâmetro é DOUBLE.

m_offset
Especifica o deslocamento de todas as coordenadas M de geometrias que são representadas neste sistema de referência espacial. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor para este parâmetro na definição do sistema de referência espacial não será alterado.

O deslocamento é subtraído antes do fator de escala *m_scale* ser aplicado quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender.

O tipo de dados deste parâmetro é DOUBLE.

m_scale
Especifica o fator de escala de todas as coordenadas M de geometrias que são representadas neste sistema de referência espacial. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor para este parâmetro na definição do sistema de referência espacial não será alterado.

O fator de escala é aplicado (multiplicação) após o deslocamento *m_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender.

O tipo de dados deste parâmetro é DOUBLE.

coordsys_name
Identifica exclusivamente o sistema de coordenadas no qual se baseia este sistema de referência espacial. O sistema de coordenadas deve estar listado na exibição ST_COORDINATE_SYSTEMS. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o sistema de coordenadas que é utilizado para este sistema de referência espacial não será alterado.

O valor de *coordsys_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

description
Descreve o sistema de referência espacial, explicando sua finalidade. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, as informações de descrição sobre o sistema de referência espacial não será alterado.

O tipo de dados deste parâmetro é VARCHAR(256).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_alter_srs. Este exemplo utiliza um comando DB2 CALL para alterar o valor de parâmetro *description* de um sistema de referência espacial denominado SRSDEMO:

```
call db2gse.ST_alter_srs('SRSDEMO',NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,
     NULL,NULL,'SRS for GSE Demo Program: offices table',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- “ST_drop_srs” na página 251
- “ST_create_srs” na página 240

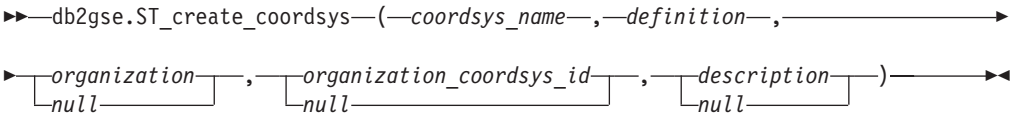
ST_create_coordsys

Utilize este procedimento armazenado para armazenar informações no banco de dados sobre um novo sistema de coordenadas. Quando este procedimento armazenado é processado, as informações sobre o sistema de coordenadas são incluídas na exibição do catálogo DB2GSE.ST_COORDINATE_SYSTEMS.

Autorização:

O ID de usuário sob o qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM.

Sintaxe:



Descrições de parâmetros:

coordsys_name
Identifica exclusivamente o sistema de coordenadas. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *coordsys_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

definition
Define o sistema de coordenadas. Você deve especificar um valor diferente de nulo para este parâmetro. O fornecedor do sistema de coordenadas geralmente possui as informações para este parâmetro.

O tipo de dados deste parâmetro é VARCHAR(2048).

organization
Nomeia a organização que definiu o sistema de coordenadas e forneceu a definição do mesmo; por exemplo, "European Petroleum Survey Group (EPSG)". Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo.

Se este parâmetro for nulo, o parâmetro *organization_coordsys_id* também deverá ser nulo. Se este parâmetro não for nulo, o parâmetro *organization_coordsys_id* não poderá ser nulo; nesse caso, a combinação dos parâmetros *organization* e *organization_coordsys_id* identifica exclusivamente o sistema de coordenadas.

O tipo de dados deste parâmetro é VARCHAR(128).

organization_coordsys_id
Especifica um identificador numérico. A entidade que é especificada no parâmetro *organization* atribui este valor. O valor não é necessariamente exclusivo entre todos sistemas de coordenadas. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo.

Se este parâmetro for nulo, o parâmetro *organization* também deverá ser nulo. Se este parâmetro não for nulo, o parâmetro *organization* não poderá ser nulo;

neste caso, a combinação dos parâmetros *organization* e *organization_coordsys_id* identifica exclusivamente o sistema de coordenadas.

O tipo de dados deste parâmetro é INTEGER.

description
Descreve o sistema de coordenadas, explicando sua finalidade. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma informação de descrição sobre o sistema de coordenadas será gravada.

O tipo de dados deste parâmetro é VARCHAR(256).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_create_coordsys. Este exemplo utiliza um comando DB2 CALL para criar um sistema de coordenadas com os seguintes valores de parâmetros:

- Parâmetro *coordsys_name*: NORTH_AMERICAN_TEST
- Parâmetro *definition*:

```
GEOGCS["GCS_North_American_1983",
DATUM["D_North_American_1983",
SPHEROID["GRS_1980",6378137.0,298.257222101]],
PRIMEM["Greenwich",0.0],
UNIT["Degree",0.0174532925199433]]
```

- Parâmetro *organization*: EPSG
- Parâmetro *organization_coordsys_id*: 1001
- Parâmetro *description*: Teste de Sistemas de Coordenadas

```
call db2gse.ST_create_coordsys('NORTH_AMERICAN_TEST',
     'GEOGCS["GCS_North_American_1983",DATUM["D_North_American_1983",
     SPHEROID["GRS_1980",6378137.0,298.257222101]],
     PRIMEM["Greenwich",0.0],UNIT["Degree",
     0.0174532925199433]]','EPSG',1001,'Teste de Sistemas de Coordenadas',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_drop_srs" na página 251
- "ST_alter_srs" na página 234

ST_create_srs

Utilize estes procedimentos armazenados para criar um sistema de referência espacial. Um sistema de referência espacial é definido pelo sistema de coordenadas, pela precisão e pelas extensões de coordenadas que são representadas no sistema de referência espacial. As extensões são os valores de coordenadas mínimo e máximo possíveis para as coordenadas X, Y, Z e M.

Internamente, o DB2 Spatial Extender armazena os valores de coordenada como inteiros positivos. Portanto, durante o cálculo, o impacto de erros de arredondamento (que são seriamente dependentes do valor real para operações de ponto flutuante) pode ser reduzido. O desempenho das operações espaciais também pode melhorar significativamente.

Este procedimento armazenado possui duas variações:

- A primeira variação utiliza os fatores de conversão (deslocamentos e fatores de escala) como parâmetros de entrada.
- A segunda variação utiliza as extensões e a precisão como parâmetros de entrada e calcula os fatores de conversão internamente.

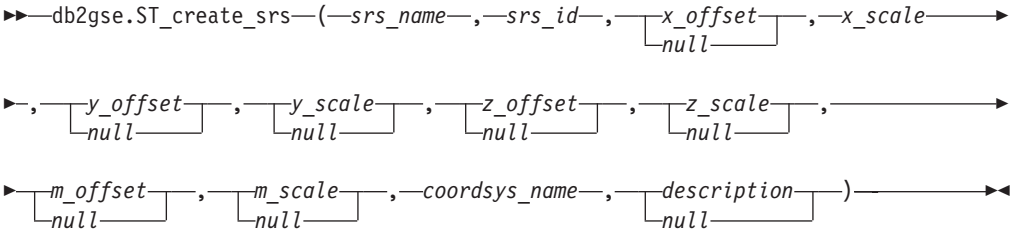
Este procedimento armazenado substitui o db2gse.gse_enable_sref.

Autorização:

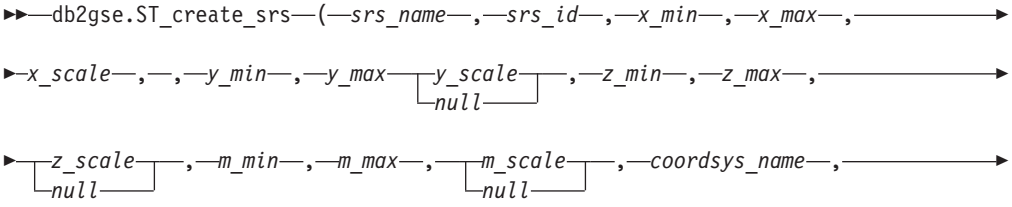
Nenhuma é necessária.

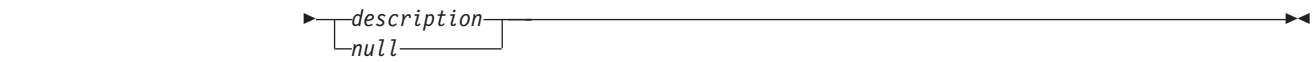
Sintaxe:

Com fatores de conversão (versão 1):



Com extensão máxima possível (versão 2):





Descrições de parâmetros:
Com fatores de conversão (versão 1):

srs_name
Identifica o sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *srs_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

srs_id
Identifica exclusivamente o sistema de referência espacial. Este identificador numérico é utilizado como um parâmetro de entrada para várias funções espaciais. Você deve especificar um valor diferente de nulo para este parâmetro.

Para um sistema de referência espacial geodésico, o valor *srs_id* deve estar no intervalo de 2000000318 a 2000001000. O DB2 Geodetic Extender fornece sistemas de referência espacial geodésicos predefinidos com valores *srs_id* de 2000000000 a 2000000317.

O tipo de dados deste parâmetro é INTEGER.

x_offset
Especifica o deslocamento de todas as coordenadas X de geometrias que são representadas neste sistema de referência espacial. O deslocamento é subtraído antes do fator de escala *x_scale* ser aplicado quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. (*WKT* significa well-kown text e*WKB* well-known binary.) Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 0 (zero) será utilizado.

O tipo de dados deste parâmetro é DOUBLE.

x_scale
Especifica o fator de escala de todas as coordenadas X de geometrias que são representadas neste sistema de referência espacial. O fator de escala é aplicado (multiplicação) após o deslocamento *x_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. O valor de *x_offset* é especificado explicitamente ou um valor padrão 0 é utilizado para *x_offset*. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

y_offset
Especifica o deslocamento de todas as coordenadas Y de geometrias que são representadas neste sistema de referência espacial. O deslocamento é subtraído antes do fator de escala *y_scale* ser aplicado quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 0 (zero) será utilizado.

O tipo de dados deste parâmetro é DOUBLE.

y_scale
Especifica o fator de escala de todas as coordenadas Y de geometrias que são representadas neste sistema de referência espacial. O fator de escala é aplicado (multiplicação) após o deslocamento *y_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. O valor de *y_offset* é especificado explicitamente ou um valor padrão 0 é utilizado para *y_offset*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor do parâmetro *x_scale* será utilizado. Se você especificar um valor diferente de nulo para este parâmetro, o valor especificado deverá corresponder ao valor do parâmetro *x_scale*.

O tipo de dados deste parâmetro é DOUBLE.

z_offset
Especifica o deslocamento de todas as coordenadas Z de geometrias que são representadas neste sistema de referência espacial. O deslocamento é subtraído antes do fator de escala *z_scale* ser aplicado quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 0 (zero) será utilizado.

O tipo de dados deste parâmetro é DOUBLE.

z_scale
Especifica o fator de escala de todas as coordenadas Z de geometrias que são representadas neste sistema de referência espacial. O fator de escala é aplicado (multiplicação) após o deslocamento *z_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. O valor de *z_offset* é especificado explicitamente ou um valor padrão 0 é utilizado para *z_offset*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 1 será utilizado.

O tipo de dados deste parâmetro é DOUBLE.

m_offset
Especifica o deslocamento de todas as coordenadas M de geometrias que são representadas neste sistema de referência espacial. O deslocamento é subtraído antes do fator de escala *m_scale* ser aplicado quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 0 (zero) será utilizado.

O tipo de dados deste parâmetro é DOUBLE.

m_scale
Especifica o fator de escala de todas as coordenadas M de geometrias que são representadas neste sistema de referência espacial. O fator de escala é aplicado (multiplicação) após o deslocamento *m_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. O valor de *m_offset* é especificado explicitamente ou um valor padrão 0 é utilizado para *m_offset*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 1 será utilizado.

O tipo de dados deste parâmetro é DOUBLE.

coordsys_name
Identifica exclusivamente o sistema de coordenadas no qual se baseia este sistema de referência espacial. O sistema de coordenadas deve estar listado na exibição ST_COORDINATE_SYSTEMS. Você deve fornecer um valor diferente de nulo para este parâmetro.

O valor de *coordsys_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

description
Descreve o sistema de referência espacial, explicando a finalidade do aplicativo. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma informação de descrição será gravada.

O tipo de dados deste parâmetro é VARCHAR(256).

Com extensão máxima possível (versão 2):

srs_name
Identifica o sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *srs_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

srs_id
Identifica exclusivamente o sistema de referência espacial. Este identificador numérico é utilizado como um parâmetro de entrada para várias funções espaciais. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é INTEGER.

x_min
Especifica o valor mínimo possível da coordenada X para todas as geometrias que utilizam este sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

x_max
Especifica o valor máximo possível da coordenada X para todas as geometrias que utilizam este sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

Dependendo do valor de *x_scale*, o valor que é mostrado na exibição DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS pode ser maior do que o valor especificado aqui. O valor da exibição está correto.

O tipo de dados deste parâmetro é DOUBLE.

x_scale
Especifica o fator de escala de todas as coordenadas X de geometrias que são representadas neste sistema de referência espacial. O fator de escala é aplicado (multiplicação) após o deslocamento *x_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma)

para a representação interna do DB2 Spatial Extender. O cálculo do deslocamento *x_offset* baseia-se no valor de *x_min*. Você deve fornecer um valor diferente de nulo para este parâmetro.

Se os parâmetros *x_scale* e *y_scale* forem especificados, os valores deverão corresponder.

O tipo de dados deste parâmetro é DOUBLE.

y_min
Especifica o valor mínimo possível da coordenada Y para todas as geometrias que utilizam este sistema de referência espacial. Você deve fornecer um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

y_max
Especifica o valor máximo possível da coordenada Y para todas as geometrias que utilizam este sistema de referência espacial. Você deve fornecer um valor diferente de nulo para este parâmetro.

Dependendo do valor de *y_scale*, o valor que é mostrado na exibição DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS pode ser maior do que o valor especificado aqui. O valor da exibição está correto.

O tipo de dados deste parâmetro é DOUBLE.

y_scale
Especifica o fator de escala de todas as coordenadas Y de geometrias que são representadas neste sistema de referência espacial. O fator de escala é aplicado (multiplicação) após o deslocamento *y_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. O cálculo do deslocamento *y_offset* baseia-se no valor de *y_min*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor do parâmetro *x_scale* será utilizado. Se os parâmetros *y_scale* e *x_scale* forem especificados, os valores deverão corresponder.

O tipo de dados deste parâmetro é DOUBLE.

z_min
Especifica o valor mínimo possível da coordenada Z para todas as geometrias que utilizam este sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

z_max
Especifica o valor máximo possível da coordenada Z para todas as geometrias que utilizam este sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

Dependendo do valor de *z_scale*, o valor que é mostrado na exibição DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS pode ser maior do que o valor especificado aqui. O valor da exibição está correto.

O tipo de dados deste parâmetro é DOUBLE.

z_scale
Especifica o fator de escala de todas as coordenadas Z de geometrias que são representadas neste sistema de referência espacial. O fator de escala é aplicado (multiplicação) após o deslocamento *z_offset* ser subtraído quando as

geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. O cálculo do deslocamento *z_offset* baseia-se no valor de *z_min*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 1 será utilizado.

O tipo de dados deste parâmetro é DOUBLE.

m_min
Especifica o valor mínimo possível da coordenada M para todas as geometrias que utilizam este sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

m_max
Especifica o valor máximo possível da coordenada M para todas as geometrias que utilizam este sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

Dependendo do valor de *m_scale*, o valor que é mostrado na exibição DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS pode ser maior do que o valor especificado aqui. O valor da exibição está correto.

O tipo de dados deste parâmetro é DOUBLE.

m_scale
Especifica o fator de escala de todas as coordenadas M de geometrias que são representadas neste sistema de referência espacial. O fator de escala é aplicado (multiplicação) após o deslocamento *m_offset* ser subtraído quando as geometrias são convertidas de representações externas (WKT, WKB, forma) para a representação interna do DB2 Spatial Extender. O cálculo do deslocamento *m_offset* baseia-se no valor de *m_min*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 1 será utilizado.

O tipo de dados deste parâmetro é DOUBLE.

coordsys_name
Identifica exclusivamente o sistema de coordenadas no qual se baseia este sistema de referência espacial. O sistema de coordenadas deve estar listado na exibição ST_COORDINATE_SYSTEMS. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *coordsys_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

description
Descreve o sistema de referência espacial, explicando a finalidade do aplicativo. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma informação de descrição será gravada.

O tipo de dados deste parâmetro é VARCHAR(256).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro,

sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_create_srs. Este exemplo utiliza um comando DB2 CALL para criar um sistema de referência espacial denominado SRSDEMO com os seguintes valores de parâmetros:

- *srs_id*: 1000000
- *x_offset*: -180
- *x_scale*: 1000000
- *y_offset*: -90
- *y_scale*: 1000000

```
call db2gse.ST_create_srs('SRSDEMO',1000000,
                    -180,1000000, -90, 1000000,
                    0, 1, 0, 1,'NORTH_AMERICAN',
                    'SRS for GSE Demo Program: customer table',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Conceitos Relacionados:

- "Sistemas de Referência Espacial" na página 67

Tarefas Relacionadas:

- "Criando um Sistema de Referência Espacial" na página 74

ST_disable_autogeocoding

Utilize este procedimento armazenado para especificar que o DB2 Spatial Extender deve parar a sincronização de uma coluna geocodificada com sua(s) coluna(s) de geocoding associada(s). Uma *coluna de geocoding* é utilizada como entrada no geocoder.

Este procedimento armazenado substitui o db2gse.gse_disable_autogc.

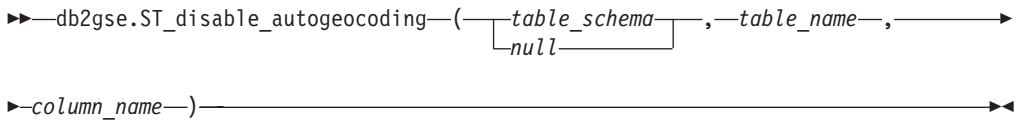
Autorização:

O ID de usuário sob o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual estão definidos os disparos que estão sendo eliminados.
- Privilégio CONTROL nesta tabela
- Privilégios ALTER e UPDATE nesta tabela

Nota: Para os privilégios CONTROL e ALTER, é necessário ter autoridade DROPIN no esquema DB2GSE.

Sintaxe:



Descrições de parâmetros:

table_schema
Nomeia o esquema ao qual pertence a tabela ou exibição especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela na qual estão definidos os disparos que você deseja eliminar. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Nomeia a coluna geocodificada que é mantida pelos disparos que você deseja eliminar. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados para este parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_disable_autogeocoding. Este exemplo utiliza um comando DB2 CALL para desativar o autogeocoding na coluna LOCALIZAÇÃO da tabela denominada CLIENTES:

```
call db2gse.ST_disable_autogeocoding(NULL,'CLIENTES','LOCALIZAÇÃO',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_enable_autogeocoding" na página 252
- "ST_setup_geocoding" na página 279

ST_disable_db

Utilize este procedimento armazenado para remover recursos que permitem ao DB2 Spatial Extender armazenar dados espaciais e suportar operações que são executadas nesses dados.

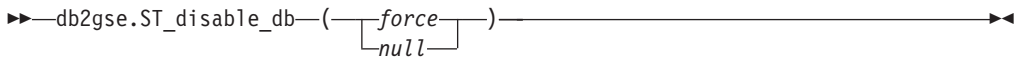
Este procedimento armazenado ajuda a resolver problemas ou questões que surgem após a ativação do banco de dados para operações espaciais. Por exemplo, você pode ativar um banco de dados para operações espaciais e, depois, decidir utilizar um outro banco de dados com o DB2 Spatial Extender. Contanto que não tenha definido colunas espaciais ou importado dados espaciais, você pode chamar este procedimento armazenado para remover todos os recursos espaciais do primeiro banco de dados. Em razão da interdependência entre colunas espaciais e as definições de tipos, não é possível eliminar as definições de tipos quando existem colunas desses tipos. Se você já tiver definido as colunas espaciais mas ainda desejar desativar um banco de dados para as operações espaciais, deverá especificar um valor diferente de 0 (zero) para o parâmetro *force* para remover todos os recursos espaciais do banco de dados que não possuem outras dependências dos mesmos.

Este procedimento armazenado substitui o db2gse.gse_disable_db.

Autorização:

O ID de usuário sob o qual este procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM no banco de dados a partir do qual os recursos do DB2 Spatial Extender devem ser removidos.

Sintaxe:



Descrições de Parâmetros:

force
Especifica que você deseja desativar um banco de dados para operações espaciais, mesmo que você tenha objetos de banco de dados que sejam dependentes dos tipos espaciais ou funções espaciais. Embora você tenha que especificar um valor para este parâmetro, o valor poderá ser nulo. Se você especificar um valor diferente de 0 (zero) ou nulo para o parâmetro *force*, o banco de dados será desativado e todos os recursos do DB2 Spatial Extender serão removidos (se possível). Se você especificar 0 (zero) ou nulo, o banco de dados não será desativado se algum objeto de banco de dados for dependente de tipos espaciais ou funções espaciais. Os objetos de banco de dados que podem ter essas dependências incluem tabelas, exibições, limitações, disparos, colunas geradas, métodos, funções, procedimentos e outros tipos de dados (subtipos ou tipos estruturados com um atributo espacial).

O tipo de dados deste parâmetro é SMALLINT.

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_disable_db. Este exemplo utiliza um comando DB2 CALL para desativar o banco de dados para operações espaciais, com o valor 1 para o parâmetro *force*:

```
call db2gse.ST_disable_db(1,?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_alter_coordsys" na página 232
- "ST_create_coordsys" na página 238

ST_drop_coordsys

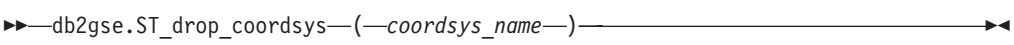
Utilize este procedimento armazenado para excluir informações sobre um sistema de coordenadas do banco de dados. Quando este procedimento armazenado é processado, as informações sobre o sistema de coordenadas são removidas da exibição do catálogo DB2GSE.ST_COORDINATE_SYSTEMS.

Restrição: Não é possível eliminar um sistema de coordenadas no qual baseia-se um sistema de referência espacial.

Autorização:

O ID de usuário sob o qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM.

Sintaxe:



Descrições de parâmetros:

coordsys_name
Identifica exclusivamente o sistema de coordenadas. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *coordsys_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_drop_coordsys. Este exemplo utiliza um comando DB2 CALL para excluir um sistema de coordenadas denominado NORTH_AMERICAN_TEST do banco de dados:

```
call db2gse.ST_drop_coordsys('NORTH_AMERICAN_TEST',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

ST_drop_srs

Use este procedimento armazenado para eliminar um sistema de referência espacial. Quando este procedimento armazenado é processado, as informações sobre o sistema de referência espacial são removidas da exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS.

Restrição: Você não pode eliminar um sistema de referência espacial se uma coluna espacial que o utiliza estiver registrada.

Importante: Utilize este procedimento armazenado com atenção. Se você utilizar este procedimento armazenado para eliminar um sistema de referência espacial, e houver dados espaciais associados a esse sistema de referência espacial, não será mais possível executar operações espaciais nos dados espaciais.

Este procedimento armazenado substitui o db2gse.gse_disable_sref.

Autorização:

O ID de usuário sob o qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM.

Sintaxe:

Descrições de parâmetros:

srs_name
Identifica o sistema de referência espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *srs_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é

retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_drop_srs. Este exemplo utiliza um comando DB2 CALL para excluir um sistema de referência espacial denominado SRSDEMO:

```
call db2gse.ST_drop_srs('SRSDEMO',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_create_srs" na página 240
- "ST_alter_srs" na página 234

ST_enable_autogeocoding

Utilize este procedimento armazenado para especificar que o DB2 Spatial Extender deve sincronizar uma coluna geocodificada com sua(s) coluna(s) de geocoding associada(s). Uma *coluna de geocoding* é utilizada como entrada no geocoder. Toda vez que os valores são inseridos ou atualizados na coluna ou colunas de geocoding, os disparos são ativados. Esses disparos chamam o geocoder associado para geocodificar os valores inseridos ou atualizados e para colocar os dados resultantes na coluna geocodificada.

Restrição: Você só pode ativar o autogeocoding nas tabelas nas quais os disparos INSERT e UPDATE podem ser criados. Conseqüentemente, não é possível ativar o autogeocoding nas exibições ou nos pseudônimos.

Pré-requisito: Antes de ativar o autogeocoding, você deve executar a etapa de configuração de geocoding chamando o procedimento armazenado ST_setup_geocoding. A etapa de configuração de geocoding especifica os valores de parâmetros de geocoder e de geocoding. Ela também identifica as colunas de geocoding que devem ser sincronizadas com as colunas geocodificadas.

Este procedimento armazenado substitui o db2gse.gse_enable_autogc.

Autorização:

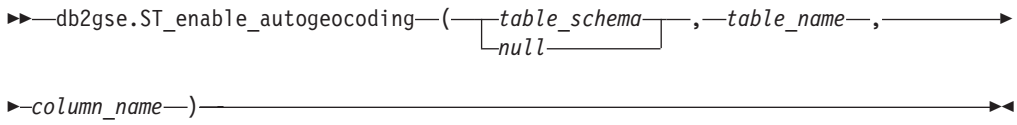
O ID de usuário sob o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual estão definidos os disparos que são criados por este procedimento armazenado
- Privilégio CONTROL na tabela
- Privilégio ALTER na tabela

Se o ID de autorização da instrução não tiver autoridade SYSADM ou DBADM, os privilégios que o ID de autorização da instrução retém (sem considerar os privilégios PUBLIC ou de grupo) devem incluir todos os seguintes privilégios, desde que exista o disparo:

- Privilégio SELECT na tabela na qual o autogeocoding está ativado
- Privilégios necessários para avaliar as expressões SQL que são especificadas para os parâmetros na configuração de geocoding

Sintaxe:



Descrições de parâmetros:

table_schema
Identifica o esquema ao qual pertence a tabela especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro for VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela que contém a coluna na qual os dados geocodificados serão inseridos ou atualizados. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro for VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Identifica a coluna na qual os dados geocodificados serão inseridos ou atualizados. Esta coluna é chamada de coluna geocodificada. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro for VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_enable_autogeocoding. Este exemplo utiliza um comando DB2 CALL para ativar o autogeocoding na coluna LOCALIZAÇÃO da tabela denominada CLIENTES:

```
call db2gse.ST_enable_autogeocoding(NULL,'CLIENTES','LOCALIZAÇÃO',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_setup_geocoding" na página 279

ST_enable_db

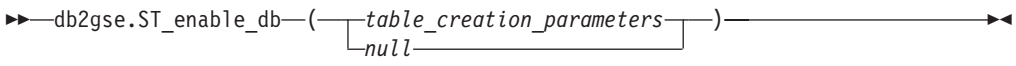
Utilize este procedimento armazenado para fornecer a um banco de dados os recursos de que ele precisa para armazenar dados espaciais e suportar operações espaciais. Estes recursos incluem tipos de dados espaciais, tipos de índices espaciais, exibições do catálogo, funções fornecidas e outros procedimentos armazenados.

Este procedimento armazenado substitui o db2gse.gse_enable_db.

Autorização:

O ID de usuário com qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM no banco de dados que está sendo ativado.

Sintaxe:



Descrições de parâmetros:

table_creation_parameters
Especifica quaisquer opções que serão incluídas nas instruções CREATE TABLE das tabelas do catálogo do DB2 Spatial Extender. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma opção será incluída nas instruções CREATE TABLE.

Para especificar essas opções, utilize a sintaxe da instrução DB2 CREATE TABLE. Por exemplo, para especificar uma área de tabela na qual as tabelas serão criadas, utilize:

```
IN tsName INDEX IN indexTsName
```

O tipo de dados deste parâmetro é VARCHAR(32K).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

O exemplo a seguir mostra como utilizar a CLI (Call Level Interface) para chamar o procedimento armazenado ST_enable_db:

```
SQLHANDLE  henv;
  SQLHANDLE  hdbc;
  SQLHANDLE  hstmt;
  SQLCHAR    uid[MAX_UID_LENGTH + 1];
  SQLCHAR    pwd[MAX_PWD_LENGTH + 1];
  SQLINTEGER ind[3];
  SQLINTEGER msg_code = 0;
  char       msg_text[1024] = "";
  SQLRETURN  rc;
  char       *table_creation_parameters = NULL;

  /* Alocar identificador de ambiente */
  rc = SQLAllocHandle(SQL_HANDLE_ENV, SQL_NULL_HANDLE, &henv);

  /* Alocar identificador de banco de dados */
  rc = SQLAllocHandle(SQL_HANDLE_DBC, henv, &hdbc);

  /* Estabelecer uma conexão com o banco de dados "testdb" */
  rc = SQLConnect(hdbc, (SQLCHAR *)"testdb",
  SQL_NTS, (SQLCHAR *)uid,SQL_NTS,
                 (SQLCHAR *)pwd, SQL_NTS);


  /* Alocar identificador de instrução */
  rc = SQLAllocHandle(SQL_HANDLE_STMT, hdbc, &hstmt) ;

  /* Associar a instrução SQL para chamar o procedimento armazenado
     ST_enable_db    */
  /* com o identificador de instrução e enviar a instrução para o DBMS para
     ser preparada. */
  rc = SQLPrepare(hstmt, "call db2gse!ST_enable_db(?,?,?)", SQL_NTS);

  /* Ligar o marcador de primeiro parâmetro na instrução de chamada de SQL */
  /* o parâmetro de entrada para os parâmetros de criação de tabela,
     à variável */
  /* table_creation_parameters.                                    */
  ind[0] = SQL_NULL_DATA;
  rc = SQLBindParameter(hstmt, 1, SQL_PARAM_OUTPUT, SQL_C_CHAR,
```

```
      SQL_VARCHAR, 255, 0, table_creation_parameters, 256, &ind[0]);

/* Ligar o marcador de segundo parâmetro na instrução de chamada de SQL,*/
/* o parâmetro de saída para código de mensagem retornado, à variável
   msg_code.*/
ind[1] = 0;
rc = SQLBindParameter(hstmt, 2, SQL_PARAM_OUTPUT, SQL_C_LONG,
                      SQL_INTEGER, 0, 0, &msg_code, 4, &ind[1]);

/* Ligar o marcador de terceiro parâmetro na instrução de
   chamada de SQL, o */
/* parâmetro de saída para texto da mensagem retornado, à
   variável msg_text.*/
ind[2] = 0;
rc = SQLBindParameter(hstmt, 3, SQL_PARAM_OUTPUT, SQL_C_CHAR,
                      SQL_VARCHAR, (sizeof(msg_text)-1), 0, msg_text,
                       sizeof(msg_text), &ind[2]);
rc = SQLExecute(hstmt);
```

Referência Relacionada:

- "ST_disable_db" na página 248

ST_export_shape

Utilize este procedimento armazenado para exportar uma coluna espacial e sua tabela associada a um arquivo de formas.

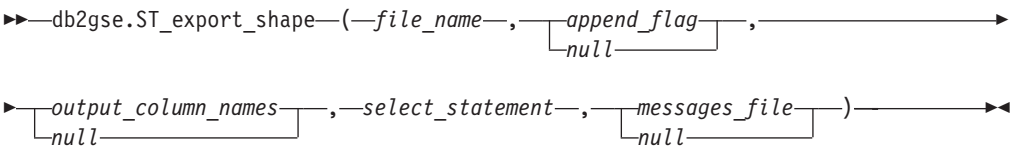
Este procedimento armazenado substitui o db2gse.gse_export_shape.

Autorização:

O ID de usuário sob o qual este procedimento armazenado é chamado deve ter os privilégios necessários para executar com êxito a instrução SELECT a partir da qual os dados serão exportados.

O procedimento armazenado, que é executado como um processo que pertence ao proprietário do DB2 instance, deve ter os privilégios necessários na máquina do servidor para criar ou gravar nos arquivos de formas.

Sintaxe:



Descrições de parâmetros:

file_name
Especifica o nome completo do caminho de um arquivo de formas para o qual os dados especificados serão exportados. Você deve especificar um valor diferente de nulo para este parâmetro.

Você pode utilizar o procedimento armazenado ST_export_shape para exportar um novo arquivo ou para exportar para um arquivo existente, anexando os dados exportados ao mesmo:

- Se você estiver exportando para um novo arquivo, poderá especificar a extensão do arquivo opcional como .shp ou .SHP. Se você especificar .shp ou

.SHP para a extensão do arquivo, o DB2 Spatial Extender criará o arquivo com o valor *file_name* especificado. Se você não especificar a extensão do arquivo opcional, o DB2 Spatial Extender criará o arquivo que possui o nome do valor *file_name* especificado e com uma extensão .shp.

- Se você estiver exportando dados anexando os dados a um arquivo existente, o DB2 Spatial Extender primeiro procurará uma correspondência exata do nome especificado para o parâmetro *file_name*. Se o DB2 Spatial Extender não encontrar uma correspondência exata, ele primeiro procurará um arquivo com a extensão .shp e, em seguida, um arquivo com a extensão .SHP.

Se o valor do parâmetro *append_flag* indicar que você não está anexando a um arquivo existente, mas o arquivo nomeado no parâmetro *file_name* já existir, o DB2 Spatial Extender retornará um erro e não irá sobrepor o arquivo.

Consulte "Notas sobre utilização" na página 258 para obter uma lista de arquivos que são gravados na máquina do servidor. O procedimento armazenado, que é executado como um processo que pertence ao proprietário do DB2 instance, deve ter os privilégios necessários na máquina do servidor para criar ou gravar nos arquivos.

O tipo de dados deste parâmetro é VARCHAR(256).

append_flag
Indica se os dados que devem ser exportados serão anexados a um arquivo de formas existente. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Indique se você deseja anexar a um arquivo de formas existente da seguinte maneira:

- Se você desejar anexar dados a um arquivo de formas existente, especifique qualquer valor diferente de 0 (zero) e nulo. Nesse caso, a estrutura de arquivos deve corresponder aos dados exportados; caso contrário, um erro será retornado.
- Se você desejar exportar para um novo arquivo, especifique 0 (zero) ou nulo. Nesse caso, o DB2 Spatial Extender não sobrepõe os arquivos existentes.

O tipo de dados deste parâmetro é SMALLINT.

output_column_names
Especifica um ou mais nomes de colunas (separados por vírgulas) que serão utilizados para colunas não-espaciais no arquivo dBASE de saída. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, serão utilizados os nomes derivados da instrução SELECT.

Se você especificar este parâmetro, mas não colocar os nomes de colunas entre aspas duplas, os nomes de colunas serão convertidos para maiúsculas. O número de colunas especificadas deve corresponder ao número de colunas retornadas da instrução SELECT, conforme especificado no parâmetro *select_statement*, excluindo a coluna espacial.

O tipo de dados deste parâmetro é VARCHAR(32K).

select_statement
Especifica a subseleção que retorna os dados a serem exportados. A subseleção deve referir-se exatamente a uma coluna espacial e a qualquer número de colunas de atributo. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é VARCHAR(32K).

messages_file
Especifica o nome completo do caminho do arquivo (na máquina do servidor) que conterá as mensagens sobre a operação de exportação. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhum arquivo para as mensagens do DB2 Spatial Extender será criado.

As mensagens que são enviadas para este arquivo de mensagens podem ser:

- Mensagens informativas como, por exemplo, um resumo da operação de exportação
- Mensagens de erro de dados que não puderam ser exportados, por exemplo, por causa de sistemas de coordenadas diferentes

O procedimento armazenado, que é executado como um processo que pertence ao proprietário da instância do DB2, deve ter os privilégios necessários no servidor para criar o arquivo.

O tipo de dados deste parâmetro é VARCHAR(256).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Notas de Uso:

Você pode exportar apenas uma coluna espacial por vez.

O procedimento armazenado ST_export_shape cria ou grava nos quatro arquivos a seguir:

- O arquivo de formas principal (extensão .shp).
- O arquivo de índice de formas (extensão .shx).
- Um arquivo dBASE que contém dados de colunas não-espaciais (extensão .dbf). Este arquivo é criado apenas se as colunas de atributo realmente precisarem ser exportadas
- Um arquivo de projeção que especifica o sistema de coordenadas que está associado aos dados espaciais, se o sistema de coordenadas não for igual a "UNSPECIFIED" (extensão .prj). O sistema de coordenadas é obtido do primeiro registro espacial. Ocorrerá um erro se registros subseqüentes tiverem sistemas de coordenadas diferentes.

A tabela a seguir descreve como os tipos de dados DB2 são armazenados nos arquivos de atributos do dBASE. Todos os outros tipos de dados DB2 não são suportados.

Tabela 29. Armazenamento de Tipos de Dados DB2 nos Arquivos de Atributos

Tipo de SQL	Tipo de .dbf	Comprimento do .dbf	Decimais do .dbf	Comentários
SMALLINT	N	6	0	
INTEGER	N	11	0	
BIGINT	N	20	0	
DECIMAL	N	precision+2	escala	
REAL FLOAT(1) até FLOAT(24)	F	14	6	
DOUBLE FLOAT(25) até FLOAT(53)	F	19	9	
CHARACTER, VARCHAR, LONG VARCHAR e DATALINK	C	*len*	0	comprimento ≤ 255
DATE	D	8	0	
TIME	C	8	0	
TIMESTAMP	C	26	0	

Todos os sinônimos de tipos de dados e tipos distintos que baseiam-se nos tipos listados na tabela precedente são suportados.

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_export_shape. Este exemplo utiliza um comando DB2 CALL para exportar todas as linhas da tabela CLIENTES para um arquivo de formas que será criado e nomeado `/tmp/export_file`:

```
call db2gse.ST_export_shape('/tmp/export_file',0,NULL,
     'select * from customers','/tmp/export_msg',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

• "ST_import_shape" na página 259

ST_import_shape

Utilize este procedimento armazenado para importar um arquivo de formas para um banco de dados que será ativado para operações espaciais. O procedimento armazenado pode operar de uma das duas formas, com base no parâmetro *create_table_flag*:

- O DB2 Spatial Extender pode criar uma tabela que possui uma coluna espacial e colunas de atributos e, em seguida, pode carregar as colunas da tabela com os dados do arquivo.

- Caso contrário, os formato de dados e de atributos poderão ser carregados em uma tabela existente que tenha uma coluna espacial e colunas de atributo que correspondam aos dados do arquivo.

Este procedimento armazenado substitui o db2gse.gse_import_shape.

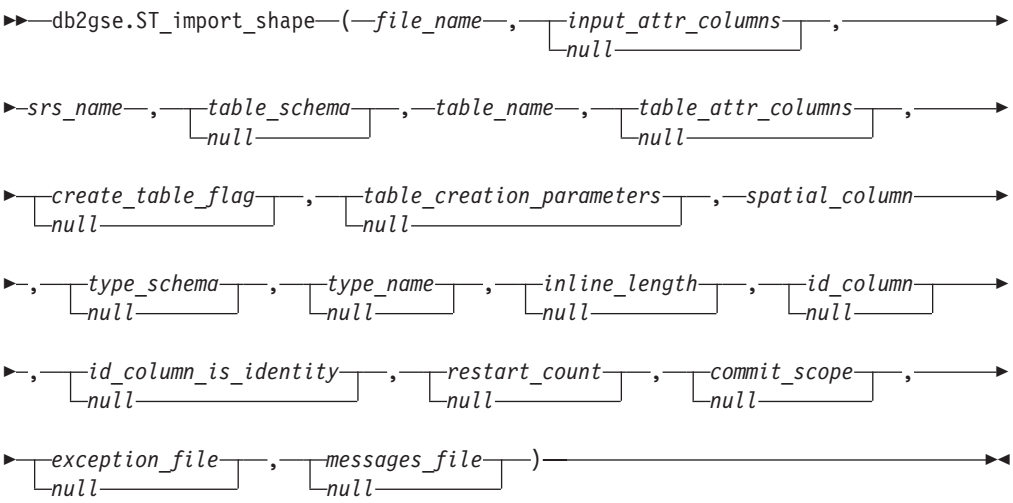
Autorização:

O proprietário da instância do DB2 deve ter os privilégios necessários na máquina do servidor para ler os arquivos de entrada e, opcionalmente, gravar os arquivos de erros. Requisitos de autorização adicionais variam dependendo se você está importando para uma tabela existente ou para uma nova tabela.

- **Ao importar para uma tabela existente**, o ID de usuário sob o qual este procedimento armazenado é chamado deve conter uma das seguintes autoridades ou privilégios:
  - SYSADM ou DBADM
  - Privilégio CONTROL na tabela ou exibição
  - Privilégio INSERT e SELECT na tabela ou exibição
- **Ao importar para uma nova tabela**, o ID de usuário sob o qual este procedimento armazenado é chamado deve conter uma das seguintes autoridades ou privilégios:
  - SYSADM ou DBADM
  - Autoridade CREATETAB no banco de dados

  O ID de usuário também deve ter uma das seguintes autoridades:
  - Autoridade IMPLICIT_SCHEMA sobre o banco de dados, se o nome do esquema da tabela não existir
  - Privilégio CREATEIN sobre o esquema, se o esquema da tabela existir

Sintaxe:



Descrições de parâmetros:

file_name
Especifica o nome completo do caminho do arquivo de formas que será importado. Você deve especificar um valor diferente de nulo para este parâmetro.

Se você especificar a extensão do arquivo opcional, especifique .shp ou .SHP. O DB2 Spatial Extender primeiro procura uma correspondência exata do nome do arquivo especificado. Se o DB2 Spatial Extender não encontrar uma correspondência exata, ele primeiro procurará um arquivo com a extensão .shp e, em seguida, um arquivo com a extensão .SHP.

Consulte "Notas sobre utilização" na página 266 para obter uma lista de arquivos requeridos, que devem residir na máquina do servidor. O procedimento armazenado, que é executado como um processo que pertence ao proprietário da instância do DB2, deve ter os privilégios necessários no servidor para ler os arquivos.

O tipo de dados deste parâmetro é VARCHAR(256).

input_attr_columns
Especifica uma lista de colunas de atributo a serem importadas do arquivo dBASE. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, todas as colunas serão importadas. Se o arquivo dBASE não existir, este parâmetro deverá ser uma cadeia vazia ou nulo.

Para especificar um valor diferente de nulo para este parâmetro, utilize uma das seguintes especificações:

- **Liste os nomes de colunas de atributo.** O exemplo a seguir mostra como especificar uma lista dos nomes das colunas de atributos que serão importadas do arquivo dBASE:

  ```
  N(COLUMN1,COLUMN5,COLUMN3,COLUMN7)
  ```

  Se um nome de coluna não for colocado entre aspas duplas, ele será convertido para maiúsculas. Cada nome na lista deve ser separado por uma vírgula. Os nomes resultantes devem corresponder exatamente aos nomes de coluna no arquivo dBASE.
- **Liste os números de colunas de atributo.** O exemplo a seguir mostra como especificar uma lista dos números das colunas de atributos que serão importadas do arquivo dBASE:

  ```
  P(1,5,3,7)
  ```

  As colunas são enumeradas iniciando com 1. Cada número na lista deve ser separado por uma vírgula.
- **Especifique que os dados de atributos não serão importados.** Especifique `""`, que é uma cadeia vazia que especifica explicitamente que o DB2 Spatial Extender *não* importará dados de atributos.

O tipo de dados deste parâmetro é VARCHAR(32K).

srs_name
Identifica o sistema de referência espacial que será utilizado para as geometrias que são importadas para a coluna espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

A coluna espacial não será registrada. O SRS (sistema de referência espacial) deve existir antes dos dados serem importados. O processo de importação não cria implicitamente o SRS, mas compara o sistema de coordenadas do SRS com o sistema de coordenadas especificado no arquivo .prj (se disponível com o arquivo de formas). O processo de importação também verifica se as extensões dos dados no arquivo de formas podem ser representadas no sistema de

referência espacial especificado. Ou seja, o processo de importação verifica se as extensões estão dentro das coordenadas X, Y, Z e M mínimas e máximas possíveis do SRS.

O valor de *srs_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_schema
Nomeia o esquema ao qual pertence a tabela especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela na qual o arquivo de formas importado será carregado. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_attr_columns
Especifica os nomes de colunas da tabela nos quais os dados de atributos do arquivo dBASE serão armazenados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, os nomes das colunas no arquivo dBASE serão utilizados.

Se este parâmetro for especificado, o número de nomes deverá corresponder ao número de colunas que são importadas do arquivo dBASE. Se a tabela existir, as definições de colunas deverão corresponder aos dados de entrada. Consulte "Notas sobre utilização" na página 266 para obter uma explicação de como os tipos de dados de atributos são mapeados para os tipos de dados DB2.

O tipo de dados deste parâmetro é VARCHAR(32K).

create_table_flag
Especifica se o processo de importação criará uma nova tabela. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo ou qualquer outro valor diferente de 0 (zero), uma nova tabela será criada. (Se a tabela já existir, um erro será retornado.) Se este parâmetro for 0 (zero), nenhuma tabela será criada, e a tabela já deverá existir.

O tipo de dados deste parâmetro é INTEGER.

table_creation_parameters
Especifica quaisquer opções a serem incluídas na instrução CREATE TABLE que cria uma tabela na qual os dados serão importados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma opção será incluída na instrução CREATE TABLE.

Para especificar quaisquer opções CREATE TABLE, utilize a sintaxe da instrução DB2 CREATE TABLE. Por exemplo, para especificar uma área de tabela na qual as tabelas serão criadas, especifique:

```
IN tsName INDEX IN indexTsName LONG IN longTsName
```

O tipo de dados deste parâmetro é VARCHAR(32K).

spatial_column
Nome da coluna espacial na tabela para a qual os formato de dados serão carregados. Você deve especificar um valor diferente de nulo para este parâmetro.

Para uma nova tabela, este parâmetro especifica o nome da nova coluna espacial que será criada. Caso contrário, este parâmetro especificará o nome de uma coluna espacial existente na tabela.

O valor de *spatial_column* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

type_schema
Especifica o nome do esquema do tipo de dados espacial (especificado pelo parâmetro *type_name*) que será utilizado ao criar uma coluna espacial em uma nova tabela. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor DB2GSE será utilizado.

O valor de *type_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

type_name
Nomeia o tipo de dados que será utilizado para os valores espaciais. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o tipo de dados será determinado pelo arquivo de formas e será um dos seguintes tipos:

- ST_Point
- ST_MultiPoint
- ST_MultiLineString
- ST_MultiPolygon

Observe que os arquivos de formas, por definição, permitem uma distinção apenas entre pontos e multipontos, mas não entre polígonos e multipolígonos ou entre cadeias de linhas e cadeias de linhas múltiplas.

Se você estiver importando para uma tabela que ainda não existe, este tipo de dados também será utilizado para o tipo de dados da coluna espacial. Nesse caso, o tipo de dados também pode ser um supertipo ST_Point, ST_MultiPoint, ST_MultiLineString ou ST_MultiPolygon.

O valor de *type_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

inline_length
Especifica, para uma nova tabela, o número máximo de bytes que serão alocados para a coluna espacial dentro da tabela. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma opção INLINE LENGTH explícita será utilizada na instrução CREATE TABLE e os padrões do DB2 serão utilizados implicitamente.

Os registros espaciais que excedem este tamanho são armazenados separadamente na área de tabela LOB, cujo acesso pode ser mais lento.

Os tamanhos típicos que são necessários para vários tipos espaciais são os seguintes:

- **Um ponto:** 292.
- **Multiponto, linha ou polígono:** O maior valor possível. Considere que o número total de bytes em uma linha não deve exceder o limite do tamanho da página da área de tabela para a qual a tabela é criada.

Consulte a documentação do DB2 sobre a instrução SQL CREATE TABLE para uma descrição completa desse valor. Consulte também o utilitário db2dart para determinar o número de geometrias em linha para as tabelas existentes e a capacidade para alterar o comprimento em linha.

O tipo de dados deste parâmetro é INTEGER.

id_column
Nomeia uma coluna que será criada para conter um número exclusivo para cada linha de dados. (As ferramentas ESRI requerem uma coluna denominada SE_ROW_ID.) Os valores exclusivos para essa coluna são gerados automaticamente durante o processo de importação. Embora você deva especificar um valor para este parâmetro, o valor poderá ser nulo se nenhuma coluna (com um ID exclusivo em cada linha) existir na tabela ou se você não estiver incluindo essa coluna em uma tabela recentemente criada. Se este parâmetro for nulo, nenhuma coluna será criada ou preenchida com números exclusivos.

Restrição: Você não pode especificar um nome de *id_column* que corresponda ao nome de alguma coluna no arquivo dBASE.

Os requisitos e o efeito deste parâmetro dependem se a tabela já existe.

- **Para uma tabela existente**, o tipo de dados do parâmetro *id_column* poderá ser qualquer tipo inteiro (INTEGER, SMALLINT ou BIGINT).
- **Para uma nova tabela que será criada**, a coluna é incluída na tabela quando o procedimento armazenado criá-la. A coluna será definida da seguinte forma:

  ```
  INTEGER NOT NULL PRIMARY KEY
  ```

  Se o valor do parâmetro *id_column_is_identity* não for nulo e não 0 (zero), a definição será expandida da seguinte forma:

  ```
  INTEGER NOT NULL PRIMARY KEY GENERATED ALWAYS AS IDENTITY
  ( START WITH 1 INCREMENT BY 1 )
  ```

O valor de *id_column* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

id_column_is_identity
Indica se a *id_column* especificada será criada utilizando a cláusula IDENTITY. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for 0 (zero) ou nulo, a coluna não será criada como a coluna de identidade. Se o parâmetro for qualquer valor diferente de 0 ou nulo, a coluna será criada como a coluna de identidade. Este parâmetro será ignorado para tabelas que já existem.

O tipo de dados deste parâmetro é SMALLINT.

restart_count
Especifica que uma operação de importação será iniciada no registro *n* + 1. Os primeiros *n* registros serão ignorados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, todos os registros (iniciando com número de registro 1) serão importados.

O tipo de dados deste parâmetro é INTEGER.

commit_scope
Especifica que um COMMIT será executado após pelo menos menos *n* registros serem importados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um valor 0 (zero) será utilizado e nenhum registro será consolidado.

O tipo de dados deste parâmetro é INTEGER.

exception_file
Especifica o nome completo do caminho de um arquivo de formas no qual são armazenados os dados de forma que não puderam ser importados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se o parâmetro for nulo, nenhum arquivo será criado.

Se você especificar um valor para o parâmetro e incluir a extensão do arquivo opcional, especifique .shp ou .SHP. Se a extensão for nula, uma extensão .shp será anexada.

O arquivo de exceção contém o bloco completo de linhas para as quais uma instrução de inserção única falhou. Por exemplo, suponha que uma linha não possa ser importada porque os formato de dados estão incorretamente codificados. Uma instrução de inserção única tenta importar 20 linhas, incluindo aquela com erro. Devido ao problema com uma única linha, o bloco inteiro de 20 linhas é gravado no arquivo de exceção.

Os registros são gravados no arquivo de exceção apenas quando esses registros podem ser corretamente identificados, como é o caso em que o tipo de registro de formas não é válido. Alguns tipos de danos nos formato de dados (arquivos .shp) e índice de formas (arquivos .shx) não permitem que os registros apropriados sejam identificados. Nesse caso, nenhum registro é gravado no arquivo de exceção e uma mensagem de erro é emitida para relatar o problema.

Se você especificar um valor para este parâmetro, quatro arquivos serão criados na máquina do servidor. Consulte “Notas sobre utilização” na página 266 para uma explicação desses arquivos. O procedimento armazenado, que é executado como um processo que pertence ao proprietário da instância do DB2, deve ter os privilégios necessários no servidor para criar os arquivos. Se os arquivos já existirem, o procedimento armazenado retornará um erro.

O tipo de dados deste parâmetro é VARCHAR(256).

messages_file
Especifica o nome completo do caminho do arquivo (na máquina do servidor)

que conterá as mensagens sobre a operação de importação. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se o parâmetro for nulo, nenhum arquivo para as mensagens do DB2 Spatial Extender será criado.

As mensagens que são gravadas no arquivo de mensagens podem ser:

- Mensagens informativas como, por exemplo, um resumo da operação de importação
- Mensagens de erro de dados que não puderam ser importados, por exemplo, por causa de sistemas de coordenadas diferentes

Estas mensagens correspondem aos formato de dados que são armazenados no arquivo de exceção (identificado pelo parâmetro *exception_file*).

O procedimento armazenado, que é executado como um processo que pertence ao proprietário da instância do DB2, deve ter os privilégios necessários no servidor para criar o arquivo. Se o arquivo já existir, o procedimento armazenado retornará um erro.

O tipo de dados deste parâmetro é VARCHAR(256).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Notas de Uso:

O procedimento armazenado ST_import_shape utiliza de um a quatro arquivos:

- O arquivo de formas principal (extensão .shp). Este arquivo é obrigatório.
- O arquivo de índice de formas (extensão .shx). Este arquivo é opcional. Se ele existir, o desempenho da operação de importação poderá melhorar.
- Um arquivo dBASE que contém dados de atributo (extensão .dbf). Este arquivo é obrigatório apenas se os dados de atributo vão ser importados.
- O arquivo de projeção que especifica o sistema de coordenadas dos formato de dados (extensão .prj). Este arquivo é opcional. Se ele existir, o sistema de coordenadas definido nele será comparado com o sistema de coordenadas do sistema de referência espacial especificado pelo parâmetro *srs_id*.

A tabela a seguir descreve como os tipos de dados de atributo dBASE são mapeados para os tipos de dados DB2. Todos os outros tipos de dados de atributo não são suportados.

Tabela 30. Relacionamento Entre os Tipos de Dados DB2 e os Tipos de Dados do Atributo dBASE

Tipo .dbf	Comprimento do .dbf (Consultar nota)	Decimais do .dbf (Consultar nota)	Tipo SQL	Comentários
N	< 5	0	SMALLINT	
N	< 10	0	INTEGER	
N	< 20	0	BIGINT	
N	*len*	*dec*	DECIMAL(*len*,*dec*)	*len*<32
F	*len*	*dec*	REAL	*len* + *dec* < 7
F	*len*	*dec*	DOUBLE	
C	*len*		CHAR(*len*)	
L			CHAR(1)	
D			DATE	

Nota: Esta tabela inclui as variáveis a seguir, ambas são definidas no cabeçalho do arquivo dBASE:

- *len*, que representa o comprimento total da coluna no arquivo dBASE. O DB2 Spatial Extender utiliza este valor para duas finalidades:
  - Definir a precisão do tipo de dados SQL DECIMAL ou o comprimento do tipo de dados SQL CHAR
  - Determinar qual dos tipos de inteiro ou ponto flutuante devem ser utilizados
- *dec*, que representa o número máximo de dígitos à direita do ponto decimal da coluna no arquivo dBASE. O DB2 Spatial Extender utiliza este valor para definir a escala para o tipo de dados SQL DECIMAL.

Por exemplo, suponha que o arquivo dBASE contenha uma coluna de dados cujo comprimento (*len*) esteja definido como 20. Suponha que o número de dígitos à direita do ponto decimal (*dec*) seja definido como 5. Quando o DB2 Spatial Extender importa dados dessa coluna, ele utiliza os valores de *len* e *dec* para derivar o seguinte tipo de dados SQL: DECIMAL(20,5).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_import_shape. Este exemplo utiliza um comando DB2 CALL para importar um arquivo de formas denominado `/tmp/officesShape` para a tabela denominada ESCRITÓRIOS:

```
call db2gse.ST_import_shape('/tmp/officesShape',NULL,'USA_SRS_1',NULL,
                     'ESCRITÓRIOS',NULL,0,NULL,'LOCATION',NULL,NULL,NULL,NULL,
                     NULL,NULL,NULL,NULL,'/tmp/import_msg',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_export_shape" na página 256

ST_register_geocoder

Utilize este procedimento armazenado para registrar um geocoder diferente de DB2SE_USA_GEOCODER, que é fornecido com o DB2 Spatial Extender. O geocoder DB2SE_USA_GEOCODER é registrado pelo DB2 Spatial Extender quando o banco de dados é ativado.

Pré-requisitos: Antes de registrar um geocoder:

- Assegure-se de que a função que implementa o geocoder já esteja criada. Cada função do geocoder pode ser registrada como um geocoder com um nome de geocoder identificado exclusivamente.
- Obtenha informações do fornecedor do geocoder, tais como:
  - A instrução SQL que cria a função
  - Os valores a serem utilizados com os parâmetros ST_create_srs para que dados geométricos possam ser suportados
  - Informações para registrar o geocoder, tais como:
    - Uma descrição do geocoder
    - Descrições dos parâmetros para o geocoder
    - Os valores padrão dos parâmetros de geocoder

O tipo de retorno da função do geocoder deve corresponder ao tipo de dados da coluna geocodificada. Os parâmetros de geocoding podem ser um nome de coluna (denominado *coluna de geocoding*) que contém os dados necessários ao geocoder. Por exemplo, os parâmetros do geocoder podem identificar endereços ou um valor de significado específico para o geocoder, tal como o score mínimo de correspondência. Se o parâmetro de geocoding for um nome de coluna, a coluna deverá estar na mesma tabela ou exibição que a coluna geocodificada.

O tipo de retorno da função do geocoder serve como o tipo de dados para a coluna geocodificada. O tipo de retorno pode ser qualquer tipo de dados DB2, tipo definido pelo usuário ou tipo estruturado. Se um tipo definido pelo usuário ou estruturado for retornado, a função do geocoder será responsável por retornar um valor válido do respectivo tipo de dados. Se a função do geocoder retornar valores de um tipo espacial, ou seja, ST_Geometry ou um de seus subtipos, a função do geocoder será responsável por construir uma geometria válida. A geometria deve ser representada utilizando um sistema de referência espacial existente. A geometria será válida se você chamar a função espacial ST_IsValid na geometria e um valor 1 for retornado. Os dados retornados da função do geocoder são atualizados ou inseridos na coluna geocodificada, dependendo de qual operação (INSERT ou UPDATE) causou a geração do valor geocodificado.

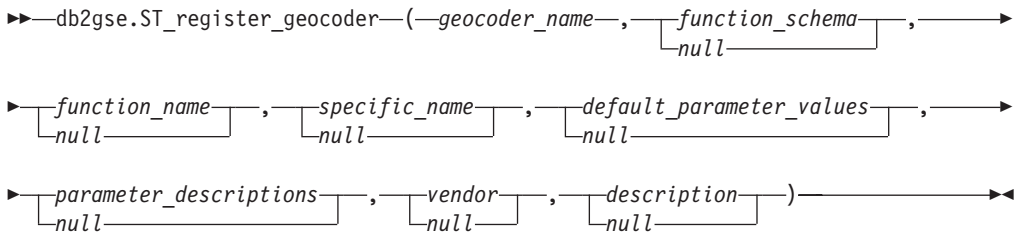
Para descobrir se um geocoder já está registrado, examine a exibição do catálogo DB2GSE.ST_GEOCODERS.

Este procedimento armazenado substitui o db2gse.gse_register_gc.

Autorização:

O ID de usuário com qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM no banco de dados que contém o geocoder que este procedimento armazenado registra.

Sintaxe:



Descrições de parâmetros:

geocoder_name
Identifica exclusivamente o geocoder. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *geocoder_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

function_schema
Nomeia o esquema para a função que implementa este geocoder. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a função.

O valor de *function_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

function_name
Especifica o nome não-qualificado da função que implementa este geocoder. A função já deve estar criada e listada em SYSCAT.ROUTINES.

Para este parâmetro, você pode especificar nulo se o parâmetro *specific_name* for especificado. Se o parâmetro *specific_name* não for especificado, o valor *function_name*, juntamente com o valor *function_schema*, implicitamente ou explicitamente definido, deverá identificar exclusivamente a função. Se o parâmetro *function_name* não for especificado, o DB2 Spatial Extender recuperará o valor *function_name* da exibição do catálogo SYSCAT.ROUTINES.

O valor *function_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

specific_name
Identifica o nome específico da função que implementa o geocoder. A função já deve estar criada e listada em SYSCAT.ROUTINES.

Para este parâmetro, você pode especificar nulo se o parâmetro *function_name* for especificado e a combinação de *function_schema* e *function_name* identificar exclusivamente a função do geocoder. Se o nome da função do geocoder estiver sobrecarregada, o parâmetro *specific_name* não poderá ser nulo. (Um nome de função está *sobrecarregado* se tiver o mesmo nome, mas não os mesmos parâmetros ou tipos de dados de parâmetros, como uma ou mais funções diferentes.)

O valor de *specific_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

default_parameter_values
Especifica a lista de valores de parâmetros padrão de geocoding para a função do geocoder. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se o parâmetro *default_parameter_values* inteiro for nulo, todos os valores de parâmetros padrão serão nulos.

Se você especificar quaisquer valores de parâmetros, especifique-os na ordem em que foram definidos pela função e separe-os com uma vírgula. Por exemplo:

```
default_parm1_value,default_parm2_value,...
```

Cada valor de parâmetro é uma expressão SQL. Siga estas diretrizes:

- Se um valor for uma cadeia, coloque-a entre aspas simples.
- Se um valor de parâmetro for um número, não coloque-o entre aspas simples.
- Se o valor do parâmetro for nulo, faça a coerção para o tipo correto. Por exemplo, em vez de especificar apenas NULL, especifique:

  ```
  CAST(NULL AS INTEGER)
  ```
- Se o parâmetro de geocoding for para uma coluna de geocoding, não especifique o valor de parâmetro padrão.

Se algum valor de parâmetro não for especificado (ou seja, se você especificar duas vírgulas consecutivas (`...,,...`)), este parâmetro deverá ser especificado quando o geocoding for configurado ou quando o geocoding for executado no modo batch com o parâmetro *parameter_values* dos respectivos procedimentos armazenados.

O tipo de dados deste parâmetro é VARCHAR(32K).

parameter_descriptions
Especifica a lista de descrições de parâmetros de geocoding para a função do geocoder. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo.

Se o parâmetro *parameter_descriptions* inteiro for nulo, todas as descrições de parâmetros serão nulas. Cada descrição de parâmetro que você especifica explica o significado e uso do parâmetro e pode conter até 256 caracteres. As descrições para os parâmetros devem ser separadas por vírgulas e devem aparecer na ordem dos parâmetros, conforme definido pela função. Se uma vírgula precisa ser utilizada na descrição de um parâmetro, coloque a cadeia entre aspas simples ou duplas. Por exemplo:

```
description,'description2, which contains a comma',description3
```

O tipo de dados deste parâmetro é VARCHAR(32K).

vendor
Nomeia o fornecedor que implementou o geocoder. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma informação sobre o fornecedor que implementou o geocoder será gravada.

O tipo de dados deste parâmetro é VARCHAR(128).

description
Descreve o geocoder, explicando sua finalidade. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma informação de descrição sobre o geocoder será gravada.

Recomendação: Inclua as seguintes informações:

- Nome do sistema de coordenadas, se dados espaciais, como WKT (Well-Known Text) ou WKB (Well-Known Binary) forem retornados
- Sistema de referência espacial, se ST_Geometry ou qualquer um de seus subtipos forem retornados
- Nome da área geográfica à qual este geocoder se aplica
- Quaisquer outras informações sobre o geocoder que os usuários devam saber

O tipo de dados deste parâmetro é VARCHAR(256).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo supõe que você deseja criar um geocoder que utiliza latitude e longitude como entrada e geocodifique para os dados espaciais ST_Point. Para fazer isso, primeiro crie uma função denominada lat_long_gc_func. Em seguida, registre um geocoder denominado SAMPLEGC, que utiliza a função lat_long_gc_func.

Segue um exemplo da instrução SQL que cria a função lat_long_gc_func, a qual retorna ST_Point:

```
CREATE FUNCTION lat_long_gc_func(latitude double,
   longitude double, srId integer)
   RETURNS db2gse.ST_Point
   LANGUAGE SQL
   RETURN db2gse.ST_Point(latitude, longitude, srId)
```

Depois que a função é criada, você pode registrá-la como um geocoder. Este exemplo mostra como utilizar o comando CALL do processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_register_geocoder para registrar um geocoder denominado SAMPLEGC, com função lat_long_gc_func:

```
call db2gse.ST_register_geocoder ('SAMPLEGC',NULL,'LAT_LONG_GC_FUNC',',,1'
            ,NULL,'My Company','Latitude/Longitude to
            ST_Point Geocoder'?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_unregister_geocoder" na página 282

ST_register_spatial_column

Utilize este procedimento armazenado para registrar uma coluna espacial e associar um SRS (sistema de referência espacial) a ele. Quando este procedimento armazenado é processado, as informações sobre a coluna espacial que está sendo registrada são incluídas na exibição do catálogo DB2GSE.ST_GEOMETRY_COLUMNS. Registrar uma coluna espacial cria uma limitação na tabela, se possível, para assegurar que todas as geometrias utilizem o SRS especificado.

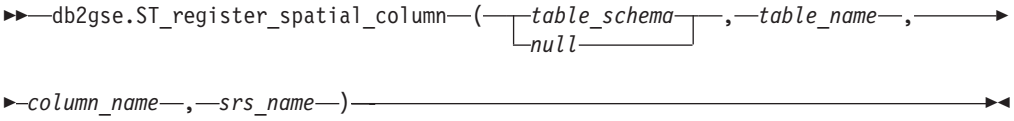
Este procedimento armazenado substitui o db2gse.gse_register_layer.

Autorização:

O ID de usuário sob o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela à qual pertence a coluna espacial que está sendo registrada
- Privilégio CONTROL ou ALTER nesta tabela

Sintaxe:



Descrições de parâmetros:

table_schema
Nomeia o esquema ao qual pertence a tabela ou exibição especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela ou exibição que contém a coluna que está sendo registrada. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Nomeia a coluna que está sendo registrada. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

srs_name
Nomeia o sistema de referência espacial que será utilizado para esta coluna espacial. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *srs_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_register_spatial_column. Este exemplo utiliza um comando DB2 CALL para registrar a coluna espacial denominada LOCALIZAÇÃO na tabela denominada CLIENTES. Este comando CALL especifica o valor de parâmetro *srs_name* como USA_SRS_1:

```
call db2gse.ST_register_spatial_column(NULL,'CLIENTES','LOCALIZAÇÃO',
     'USA_SRS_1',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_unregister_spatial_column" na página 283

ST_remove_geocoding_setup

Utilize este procedimento armazenado para remover todas as informações de configuração de geocoding para a coluna geocodificada.

Este procedimento armazenado remove informações que estão associadas à coluna geocodificada especificada a partir das exibições do catálogo DB2GSE.ST_GEOCODING e DB2GSE.ST_GEOCODING_PARAMETERS.

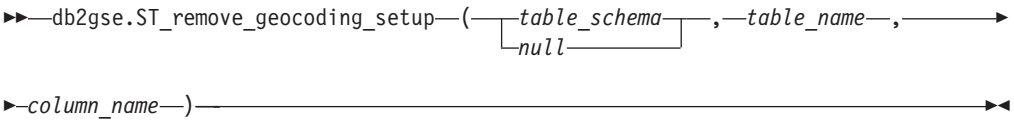
Restrição: Não é possível remover uma configuração de geocoding se autogeocoding estiver ativado para a coluna da qual foi executado geocode.

Autorização:

O ID de usuário sob o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual o geocoder especificado irá operar
- Privilégio CONTROL ou UPDATE nesta tabela

Sintaxe:



Descrições de parâmetros:

table_schema
Nomeia o esquema ao qual pertence a tabela ou exibição especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela ou exibição que contém a coluna na qual os dados geocodificados serão inseridos ou atualizados. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Nomeia a coluna na qual os dados geocodificados serão inseridos ou atualizados. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados para este parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_remove_geocoding_setup. Este exemplo utiliza um comando DB2 CALL para remover a configuração de geocoding da tabela denominada LOCALIZAÇÃO e da coluna LOCALIZAÇÃO:

```
call db2gse.ST_remove_geocoding_setup(NULL, 'CLIENTES', 'LOCALIZAÇÃO',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_setup_geocoding" na página 279

ST_run_geocoding

Utilize este procedimento armazenado para executar um geocoder no modo batch em uma coluna geocodificada.

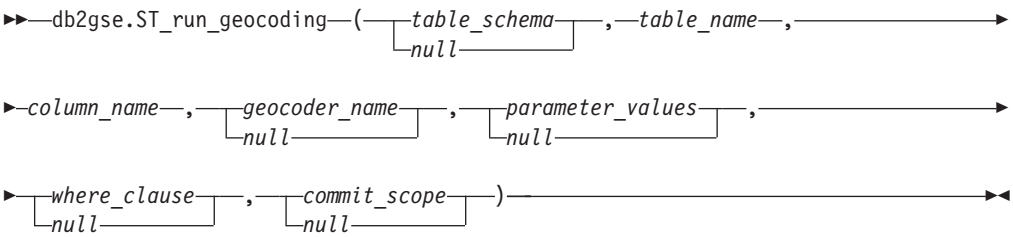
Este procedimento armazenado substitui o db2gse.gse_run_gc.

Autorização:

O ID de usuário com qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual o geocoder especificado irá operar.
- Privilégio CONTROL ou UPDATE nesta tabela

Sintaxe:



Descrições de parâmetros:

table_schema
Nomeia o esquema ao qual pertence a tabela ou exibição especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela ou exibição que contém a coluna na qual os dados geocodificados serão inseridos ou atualizados. Se um nome de exibição for especificado, a exibição deverá ser atualizável. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Nomeia a coluna na qual os dados geocodificados serão inseridos ou atualizados. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

geocoder_name
Nomeia o geocoder que executará o geocoding. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o geocoding será executado pelo geocoder que foi especificado quando o geocoding foi configurado.

O valor de *geocoder_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

parameter_values
Especifica a lista de valores de parâmetros de geocoding para a função geocoder. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se o parâmetro *parameter_values* inteiro for nulo, os valores que são utilizados são os valores de parâmetros que foram especificados quando o geocoder foi configurado os valores de parâmetros padrão do geocoder, se o geocoder não tiver sido configurado.

Se você especificar valores de parâmetros, especifique-os na ordem em que a função os definiu e separe-os com uma vírgula. Por exemplo:

```
parameter1-value,parameter2-value,...
```

Cada valor de parâmetro pode ser um nome de coluna, uma cadeia, um valor numérico ou nulo.

Cada valor de parâmetro é uma expressão SQL. Siga estas diretrizes:

- Se um valor de parâmetro for um nome de coluna de geocoding, assegure-se de que a coluna esteja na mesma tabela ou exibição na qual reside a coluna geocodificada.
- Se um valor de parâmetro for uma cadeia, coloque-a entre aspas simples.
- Se um valor de parâmetro for um número, não coloque-o entre aspas simples.
- Se o parâmetro for nulo, faça a coerção para o tipo correto. Por exemplo, em vez de especificar apenas NULL, especifique:

  ```
  CAST(NULL AS INTEGER)
  ```

Se algum valor de parâmetro não for especificado (ou seja, se você especificar duas vírgulas consecutivas (`...,,...`)), este parâmetro deverá ser especificado quando o geocoding for configurado ou quando o geocoding for executado no modo batch com o parâmetro *parameter_values* dos respectivos procedimentos armazenados.

O tipo de dados deste parâmetro é VARCHAR(32K).

where_clause
Especifica o conteúdo da cláusula WHERE, que define uma restrição do conjunto de registros que serão geocodificados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo.

Se o parâmetro *where_clause* for nulo, o comportamento resultante dependerá se o geocoding foi configurado para a coluna (especificado no parâmetro *column_name*) antes da execução do procedimento armazenado. Se o parâmetro *where_clause* for nulo, e:

- Um valor foi especificado quando o geocoding foi configurado, esse valor será utilizado para o parâmetro *where_clause*.
- O geocoding não foi configurado ou nenhum valor foi especificado quando o geocoding foi configurado, ou nenhuma cláusula foi utilizada.

Você pode especificar uma cláusula que faz referência a qualquer coluna na tabela ou exibição em que o geocoder irá operar. Não especifique a palavra-chave WHERE.

O tipo de dados deste parâmetro é VARCHAR(32K).

commit_scope
Especifica que um COMMIT será executado após cada *n* registros que são geocodificados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo.

Se o parâmetro *commit_scope* for nulo, o comportamento resultante dependerá se o geocoding foi configurado para a coluna (especificado no parâmetro *column_name*) antes da execução do procedimento armazenado. Se o parâmetro *commit_scope* for nulo e:

- Um valor foi especificado quando o geocoding foi configurado para a coluna, esse valor será utilizado para o parâmetro *commit_scope*.
- O geocoding não foi configurado ou foi configurado mas nenhum valor foi especificado, o valor padrão 0 (zero) será utilizado e nenhum COMMIT será executado.

O tipo de dados deste parâmetro é INTEGER.

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_run_geocoding. Este exemplo utiliza um comando DB2 CALL para geocodificar a coluna LOCALIZAÇÃO na tabela denominada CLIENTE. Este comando CALL especifica o valor de parâmetro *geocoder_name* como DB2SE_USA_GEOCODER e o valor de parâmetro *commit_scope* como 10. Um COMMIT será executado após cada 10 registros serem geocodificados:

```
call db2gse.ST_run_geocoding(NULL, 'CLIENTES', 'LOCALIZAÇÃO',
     'DB2SE_USA_GEOCODER',NULL,NULL,10,?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_setup_geocoding" na página 279

ST_setup_geocoding

Utilize este procedimento armazenado para associar uma coluna que será geocodificada a um geocoder e para configurar os parâmetros de geocoding correspondentes. As informações configuradas aqui são gravadas nas exibições do catálogo DB2GSE.ST_GEOCODING e DB2GSE.ST_GEOCODING_PARAMETERS.

Este procedimento armazenado não chama o geocoding. Ele fornece um modo para especificar as definições de parâmetros para a coluna que será geocodificada. Com essas definições, a chamada subseqüente de geocoding em batch ou autogeocoding pode ser feita com uma interface muito mais simples. As definições de parâmetros que são especificadas nesta etapa de configuração substituem qualquer um dos valores de parâmetros padrão do geocoder que foram especificados quando o geocoder foi registrado. Você também pode substituir estas definições de parâmetros, executando o procedimento armazenado ST_run_geocoding no modo batch.

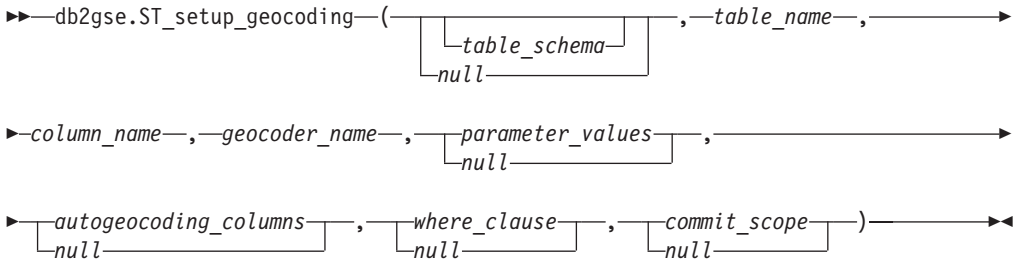
Esta etapa é um pré-requisito para autogeocoding. Você não pode ativar o autogeocoding sem antes configurar os parâmetros de geocoding. Esta etapa não é um pré-requisito para o geocoding em batch. Você pode executar o geocoding no modo batch com ou sem executar a etapa de configuração. No entanto, se a etapa de configuração for feita antes do geocoding em batch, os valores de parâmetro serão extraídos do tempo de configuração se não forem especificados no tempo de execução.

Autorização:

O ID de usuário com qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual o geocoder especificado irá operar
- Privilégio CONTROL ou UPDATE nesta tabela

Sintaxe:



Descrições de parâmetros:

table_schema

Nomeia o esquema ao qual pertence a tabela ou exibição especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela ou exibição que contém a coluna na qual os dados geocodificados serão inseridos ou atualizados. Se um nome de exibição for especificado, a exibição deverá ser atualizável. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Nomeia a coluna na qual os dados geocodificados serão inseridos ou atualizados. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

geocoder_name
Nomeia o geocoder que executará o geocoding. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *geocoder_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

parameter_values
Especifica a lista de valores de parâmetros de geocoding para a função geocoder. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se o parâmetro *parameter_values* inteiro for nulo, os valores utilizados são extraídos dos valores de parâmetros padrão no momento em que o geocoder foi registrado.

Se você especificar valores de parâmetros, especifique-os na ordem em que a função os definiu e separe-os com uma vírgula. Por exemplo:

```
parameter1-value,parameter2-value,...
```

Cada valor de parâmetro é uma expressão SQL e pode ser um nome de coluna, uma cadeia, um valor numérico ou nulo. Siga estas diretrizes:

- Se um valor de parâmetro for um nome de coluna de geocoding, assegure-se de que a coluna esteja na mesma tabela ou exibição na qual reside a coluna geocodificada.
- Se um valor de parâmetro for uma cadeia, coloque-a entre aspas simples.
- Se um valor de parâmetro for um número, não coloque-o entre aspas simples.
- Se o valor de parâmetro for especificado como um valor nulo, faça a coerção para o tipo correto. Por exemplo, em vez de especificar apenas NULL, especifique:

  ```
  CAST(NULL AS INTEGER)
  ```

Se algum valor de parâmetro não for especificado (ou seja, se você especificar duas vírgulas consecutivas (`...,,...`)), este parâmetro deverá ser especificado quando o geocoding for configurado ou quando o geocoding for executado no modo batch com o parâmetro *parameter_values* dos respectivos procedimentos armazenados.

O tipo de dados deste parâmetro é VARCHAR(32K).

autogeocoding_columns
Especifica a lista de nomes de colunas nos quais o disparo será criado. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo e o autogeocoding estiver ativado, uma atualização de qualquer coluna na tabela ativará o disparo.

Se você especificar um valor para o parâmetro *autogeocoding_columns*, especifique os nomes de colunas em qualquer ordem e separe-os com uma vírgula. O nome da coluna deve existir na mesma tabela na qual a coluna geocodificada reside.

Esta definição de parâmetro aplica-se apenas ao autogeocoding subseqüente.

O tipo de dados deste parâmetro é VARCHAR(32K).

where_clause
Especifica o conteúdo da cláusula WHERE, que define uma restrição do conjunto de registros que serão geocodificados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, nenhuma restrição será definida na cláusula WHERE.

A cláusula pode referenciar qualquer coluna na tabela ou exibição na qual o geocoder irá operar. Não especifique a palavra-chave WHERE.

Esta definição de parâmetro aplica-se apenas ao geocoding de modo batch subseqüente.

O tipo de dados deste parâmetro é VARCHAR(32K).

commit_scope
Especifica que um COMMIT será executado para cada *n* registros que serão geocodificados. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, um COMMIT será executado após todos os registros serem geocodificados.

Esta definição de parâmetro aplica-se apenas ao geocoding de modo batch subseqüente.

O tipo de dados deste parâmetro é INTEGER.

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é

retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_setup_geocoding. Este exemplo utiliza um comando DB2 CALL para configurar um processo de geocoding para a coluna geocodificada denominada LOCALIZAÇÃO na tabela CLIENTE. Este comando CALL especifica o valor de parâmetro *geocoder_name* como DB2SE_USA_GEOCODER:

```
call db2gse.ST_setup_geocoding(NULL, 'CLIENTES', 'LOCALIZAÇÃO',
    'DB2SE_USA_GEOCODER','ENDEREÇO,CIDADE,ESTADO,CEP,1,100,80,,,,"$HOME/sqllib/
     gse/refdata/ky.edg","$HOME/sqllib/samples/spatial/EDGESample.loc"',
     'ENDEREÇO,CIDADE,ESTADO,CEP',NULL,10,?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado

Referência Relacionada:

- "ST_unregister_geocoder" na página 282
- "ST_remove_geocoding_setup" na página 274

ST_unregister_geocoder

Utilize este procedimento armazenado para cancelar o registro de um geocoder diferente de DB2SE_USA_GEOCODER, que é fornecido com o DB2 Spatial Extender.

Restrição: Não é possível cancelar o registro de um geocoder se ele for especificado na configuração de geocoding para alguma coluna.

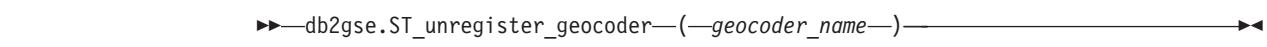
Para determinar se um geocoder está especificado na configuração de geocoding de uma coluna, verifique as exibições do catálogo DB2GSE.ST_GEOCODING e DB2GSE.ST_GEOCODING_PARAMETERS. Para procurar informações sobre o geocoder que você deseja cancelar o registro, consulte a exibição do catálogo DB2GSE.ST_GEOCODERS.

Este procedimento armazenado substitui o db2gse.gse_unregist_gc.

Autorização:

O ID de usuário com qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM no banco de dados que contém o geocoder que deverá ter o registro cancelado.

Sintaxe:



Descrições de parâmetros:

geocoder_name
Identifica exclusivamente o geocoder. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *geocoder_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_unregister_geocoder. Este exemplo utiliza um comando DB2 CALL para cancelar o registro do geocoder denominado SAMPLEGC:

```
call db2gse.ST_unregister_geocoder('SAMPLEGC',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_register_geocoder" na página 268
- "ST_setup_geocoding" na página 279

ST_unregister_spatial_column

Utilize este procedimento armazenado para remover o registro de uma coluna espacial. O procedimento armazenado remove o registro da seguinte forma:

- Removendo a associação do sistema de referência espacial com a coluna espacial. A exibição do catálogo ST_GEOMETRY_COLUMNS continua contendo a coluna espacial, mas a coluna não está mais associada a nenhum sistema de referência espacial.
- Para uma tabela base, eliminando a limitação que o DB2 Spatial Extender colocou nesta tabela para assegurar que os valores de geometria nesta coluna espacial sejam todos representados no mesmo sistema de referência espacial.

ST_unregister_spatial_column

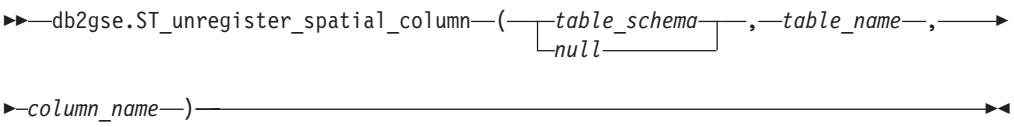
Este procedimento armazenado substitui o db2gse.gse_unregist_layer.

Autorização:

O ID de usuário com a qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM
- Privilégio CONTROL ou ALTER nesta tabela

Sintaxe:



Descrições de parâmetros:

table_schema
Nomeia o esquema ao qual pertence a tabela especificada no parâmetro *table_name*. Embora você deva especificar um valor para este parâmetro, o valor pode ser nulo. Se este parâmetro for nulo, o valor no registro especial CURRENT SCHEMA será utilizado como o nome do esquema para a tabela ou exibição.

O valor de *table_schema* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

table_name
Especifica o nome não-qualificado da tabela que contém a coluna especificada no parâmetro *column_name*. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *table_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

column_name
Nomeia a coluna espacial que você deseja cancelar o registro. Você deve especificar um valor diferente de nulo para este parâmetro.

O valor de *column_name* é convertido para maiúsculas, a menos que seja colocado entre aspas duplas.

O tipo de dados deste parâmetro é VARCHAR(128) ou, se você colocar o valor entre aspas duplas, VARCHAR(130).

Parâmetros de Saída:

msg_code
Especifica o código de mensagem que é retornado do procedimento armazenado. O valor deste parâmetro de saída identifica a condição de erro, sucesso ou aviso que foi encontrada durante o processamento do procedimento. Se este valor de parâmetro for para uma condição de sucesso ou aviso, o procedimento concluiu sua tarefa. Se o valor de parâmetro for para uma condição de erro, nenhuma alteração foi executada no banco de dados.

O tipo de dados deste parâmetro de saída é INTEGER.

msg_text
Especifica o texto da mensagem real, associado ao código de mensagem, que é retornado do procedimento armazenado. O texto da mensagem pode incluir informações adicionais sobre a condição de sucesso, aviso ou erro, tal como onde um erro foi encontrado.

O tipo de dados deste parâmetro de saída é VARCHAR(1024).

Exemplo:

Este exemplo mostra como utilizar o processador de linha de comandos do DB2 para chamar o procedimento armazenado ST_unregister_spatial_column. Este exemplo utiliza um comando DB2 CALL para cancelar o registro da coluna espacial denominada LOCALIZAÇÃO na tabela CLIENTES:

```
call db2gse.ST_unregister_spatial_column(NULL,'CLIENTES','LOCALIZAÇÃO',?,?)
```

Os dois pontos de interrogação no final deste comando CALL representam os parâmetros de saída, *msg_code* e *msg_text*. Os valores para estes parâmetros de saída são exibidos após a execução do procedimento armazenado.

Referência Relacionada:

- "ST_register_spatial_column" na página 272

Capítulo 21. Exibições do Catálogo

As exibições do catálogo do Spatial Extender contêm informações sobre:

"A Exibição do Catálogo DB2GSE.ST_COORDINATE_SYSTEMS"
Sistemas de coordenadas que você pode utilizar

"A Exibição do Catálogo DB2GSE.ST_GEOMETRY_COLUMNS" na página 288
Colunas espaciais que você pode preencher ou atualizar.

"A Exibição do Catálogo DB2GSE.ST_GEOCODERS" na página 291 e "A Exibição do Catálogo DB2GSE.ST_GEOCODING_PARAMETERS" na página 293
Geocoders que você pode utilizar

"A Exibição do Catálogo DB2GSE.ST_GEOCODING" na página 291 e "A Exibição do Catálogo DB2GSE.ST_GEOCODING_PARAMETERS" na página 293
Especificações para configurar um geocoder para ser executado automaticamente e para definir, antecipadamente, operações a serem executadas durante o geocoding de batch.

"A Exibição do Catálogo DB2GSE.ST_SIZINGS" na página 294
Comprimentos máximos permitidos para os valores que você pode atribuir a variáveis.

"A Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS" na página 295
Sistemas de referência espacial que podem ser usados.

"A Exibição do Catálogo DB2GSE.ST_UNITS_OF_MEASURE" na página 298
As unidades de medida (metros, milhas, pés e assim por diante) nas quais as distâncias geradas por funções espaciais podem ser expressas.

A Exibição do Catálogo DB2GSE.ST_COORDINATE_SYSTEMS

Consulte a exibição do catálogo DB2GSE.ST_COORDINATE_SYSTEMS para recuperar informações sobre sistemas de coordenadas registrados. O Spatial Extender registra automaticamente sistemas de coordenadas no catálogo do Spatial Extender nas seguintes situações:

- Quando ativar um banco de dados para operações espaciais.
- Quando os usuários definem sistemas de coordenadas adicionais para o banco de dados.

Para obter uma descrição de colunas nesta exibição, consulte a tabela a seguir.

Tabela 31. Colunas na Exibição do Catálogo DB2GSE.ST_COORDINATE_SYSTEMS

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
COORDSYS_NAME	VARCHAR(128)	Não	Nome deste sistema coordenado. O nome é exclusivo no banco de dados.

Tabela 31. Colunas na Exibição do Catálogo DB2GSE.ST_COORDINATE_SYSTEMS (continuação)

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
COORDSYS_TYPE	VARCHAR(128)	Não	Tipo deste sistema de coordenadas: **PROJECTED** Bidimensional. **GEOGRAPHIC** Tridimensional. Utiliza coordenadas X e Y. **GEOCENTRIC** Tridimensional. Utiliza coordenadas X, Y e Z. **UNSPECIFIED** Sistema de coordenadas abstrato ou irreal. O valor para esta coluna é obtido da coluna DEFINITION.
DEFINITION	VARCHAR(2048)	Não	Representação de texto convencional da definição deste sistema de coordenadas.
ORGANIZATION	VARCHAR(128)	Sim	Nome da organização (por exemplo, um órgão de padronizações como o European Petrol Survey Group, ou ESPG) que definiu este sistema de coordenadas. Esta coluna será nula se a coluna ORGANIZATION_COORDSYS_ID for nula.
ORGANIZATION_ COORDSYS_ID	INTEGER	Sim	Identificador numérico atribuído a este sistema de coordenadas pela organização que definiu o sistema de coordenadas. Este identificador e o valor na coluna ORGANIZATION identificam exclusivamente o sistema de coordenadas, a menos que o identificador e o valor sejam nulos. Se a coluna ORGANIZATION for nula, a coluna ORGANIZATION_COORDSYS_ID também será nula.
DESCRIPTION	VARCHAR(256)	Sim	Descrição do sistema de coordenadas que indica seu aplicativo.

A Exibição do Catálogo DB2GSE.ST_GEOMETRY_COLUMNS

Utilize a exibição do catálogo DB2GSE.ST_GEOMETRY_COLUMNS para encontrar informações sobre todas as colunas espaciais em todas as tabelas que contêm dados espaciais no banco de dados. Se uma coluna espacial foi registrada junto com um sistema de referência espacial, você também poderá utilizar a exibição para saber o nome e o identificador numérico do sistema de referência espacial. Para obter informações adicionais sobre colunas espaciais, consulte a exibição do catálogo SYSCAT.COLUMN do DB2.

Para obter uma descrição do DB2GSE.ST_GEOMETRY_COLUMNS, consulte a tabela a seguir.

Tabela 32. Colunas na Exibição do Catálogo DB2GSE.ST_GEOMETRY_COLUMNS

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
TABLE_SCHEMA	VARCHAR(128)	Não	Nome do esquema ao qual pertence a tabela que contém esta coluna espacial.

Tabela 32. Colunas na Exibição do Catálogo DB2GSE.ST_GEOMETRY_COLUMNS (continuação)

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
TABLE_NAME	VARCHAR(128)	Não	Nome não qualificado da tabela que contém esta coluna espacial.
COLUMN_NAME	VARCHAR(128)	Não	Nome desta coluna espacial. A combinação de TABLE_SCHEMA, TABLE_NAME e COLUMN_NAME que identifica exclusivamente a coluna.
TYPE_SCHEMA	VARCHAR(128)	Não	Nome do esquema ao qual pertence o tipo de dados declarado desta coluna espacial. Este nome é obtido do catálogo do DB2.
TYPE_NAME	VARCHAR(128)	Não	Nome não qualificado do tipo de dados declarado desta coluna espacial. Este nome é obtido do catálogo do DB2.
SRS_NAME	VARCHAR(128)	Sim	Nome do sistema de referência espacial que está associado a esta coluna espacial. Se nenhum sistema de referência espacial estiver associado à coluna, SRS_NAME será nulo.
SRS_ID	INTEGER	Sim	Identificador numérico do sistema de referência espacial que está associado a esta coluna espacial. Se nenhum sistema de referência espacial estiver associado à coluna, SRS_ID será nulo.

A Exibição do Catálogo DB2GSE.ST_GEOCODER_PARAMETERS

Ao ativar um banco de dados para operações espaciais, as informações sobre os parâmetros do geocoder fornecido, DB2GSE_USA_GEOCODER, são automaticamente registradas no catálogo do DB2 Spatial Extender. Se você registrar geocoders adicionais, as informações sobre seus parâmetros também serão registradas no catálogo. Para recuperar informações sobre parâmetros de geocoders do catálogo, consulte a exibição do catálogo DB2GSE.ST_GEOCODER_PARAMETERS. Para obter uma descrição de colunas nesta exibição, consulte a tabela a seguir.

Para obter mais informações sobre parâmetros de geocoders, consulte a exibição do catálogo SYSCAT.ROUTINEPARMS do DB2. Para obter uma descrição desta exibição, consulte o *SQL Reference*.

Tabela 33. Colunas em DB2GSE.ST_GEOCODER_PARAMETERS

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
GEOCODER_NAME	VARCHAR(128)	Não	Nome do geocoder ao qual este parâmetro pertence.

Tabela 33. Colunas em DB2GSE.ST_GEOCODER_PARAMETERS (continuação)

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
ORDINAL	SMALLINT	Não	A posição deste parâmetro (isto é, o parâmetro especificado na coluna PARAMETER_NAME) na assinatura da função que serve como o geocoder especificado na coluna GEOCODER_NAME. Os valores combinados nas colunas GEOCODER_NAME e ORDINAL identificam exclusivamente este parâmetro. Um registro na exibição do catálogo SYSCAT.ROUTINEPARMS do DB2 também contém informações sobre este parâmetro. Este registro contém um valor que aparece na coluna ORDINAL de SYSCAT.ROUTINEPARMS. Este valor é igual ao que aparece na coluna ORDINAL da exibição DB2GSE.ST_GEOCODER_PARAMETERS.
PARAMETER_NAME	VARCHAR(128)	Sim	Nome deste parâmetro. Se não foi especificado um nome durante a criação da função à qual este parâmetro pertence, a coluna PARAMETER_NAME será nula. O conteúdo da coluna PARAMETER_NAME é obtido do catálogo do DB2.
TYPE_SCHEMA	VARCHAR(128)	Não	Nome do esquema ao qual este parâmetro pertence. Este nome é obtido do catálogo do DB2.
TYPE_NAME	VARCHAR(128)	Não	Nome não qualificado do tipo de dados dos valores atribuídos a este parâmetro. Este nome é obtido do catálogo do DB2.
PARAMETER_DEFAULT	VARCHAR(2048)	Sim	O valor padrão que deve ser atribuído a este parâmetro. O DB2 interpretará este valor como uma expressão SQL. Se o valor estiver entre aspas, ele será transmitido para o geocoder como uma cadeia. Caso contrário, a avaliação da expressão SQL determinará qual será o tipo de dados do parâmetro quando ele for transmitido para o geocoder. Se a coluna PARAMETER_DEFAULT contiver um nulo, este valor nulo será transmitido para o geocoder. O valor padrão deve ter um valor correspondente na exibição do catálogo DB2GSE.ST_GEOCODING_PARAMETERS. Ele também pode ter um valor correspondente na entrada para o procedimento armazenado ST_run_geocoding. Se qualquer um dos valores correspondentes for diferente do valor padrão, o valor correspondente substituirá o valor padrão.
DESCRIPTION	VARCHAR(256)	Sim	Descrição do parâmetro que indica seu aplicativo.

A Exibição do Catálogo DB2GSE.ST_GEOCODERS

Ao ativar um banco de dados para operações espaciais, o geocoder fornecido, DB2GSE_USA_GEOCODER, é automaticamente registrado no catálogo do DB2 Spatial Extender. Quando desejar disponibilizar geocoders adicionais para usuários, você precisa registrar estes geocoders. Para recuperar informações sobre geocoders registrados, consulte a exibição do catálogo DB2GSE.ST_GEOCODERS. Para obter uma descrição de colunas nesta exibição, consulte a tabela a seguir.

Para obter informações sobre parâmetros de geocoders, consulte a exibição do catálogo DB2GSE.ST_GEOCODER_PARAMETERS do DB2 Spatial Extender e a exibição do catálogo SYSCAT.ROUTINEPARMS do DB2. Para obter informações sobre funções que são utilizadas como geocoders, consulte a exibição do catálogo SYSCAT.ROUTINES do DB2.

Tabela 34. Colunas na Exibição do Catálogo DB2GSE.ST_GEOCODERS

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
GEOCODER_NAME	VARCHAR(128)	Não	Nome deste geocoder. Ele é exclusivo no banco de dados.
FUNCTION_SCHEMA	VARCHAR(128)	Não	Nome do esquema ao qual pertence a função que está sendo utilizada como este geocoder.
FUNCTION_NAME	VARCHAR(128)	Não	Nome não qualificado da função que está sendo utilizada como este geocoder.
SPECIFIC_NAME	VARCHAR(128)	Não	Nome específico da função que está sendo utilizada como este geocoder. Os valores combinados de FUNCTION_SCHEMA e SPECIFIC_NAME identificam exclusivamente a função que está sendo utilizada como este geocoder.
RETURN_TYPE_SCHEMA	VARCHAR(128)	Não	Nome do esquema ao qual pertence o tipo de dados do parâmetro de saída deste geocoder. Este nome é obtido do catálogo do DB2.
RETURN_TYPE_NAME	VARCHAR(128)	Não	Nome não qualificado do tipo de dados do parâmetro de saída deste geocoder. Este nome é obtido do catálogo do DB2.
VENDOR	VARCHAR(256)	Sim	Nome do fornecedor que criou este geocoder.
DESCRIPTION	VARCHAR(256)	Sim	Descrição do geocoder que indica seu aplicativo.

A Exibição do Catálogo DB2GSE.ST_GEOCODING

Ao configurar operações de geocoding, as características particulares de suas definições serão automaticamente registradas no catálogo do DB2 Spatial Extender. Para saber quais são estas características particulares, consulte as exibições do catálogo DB2GSE.ST_GEOCODING e DB2GSE.ST_GEOCODING_PARAMETERS. A exibição do catálogo DB2GSE.ST_GEOCODING, que é descrita na tabela a seguir, contém características particulares de todas as definições; por exemplo, o número de registros que um geocoder deve processar antes de cada consolidação. A exibição do catálogo DB2GSE.ST_GEOCODING_PARAMETERS contém características particulares que são específicas de cada geocoder. Por exemplo, as configurações para o geocoder fornecido, DB2GSE_USA_GEOCODER, incluem o

grau mínimo ao qual os endereços fornecidos como entrada e os endereços reais devem corresponder para que o geocoder efetue geocode na entrada. Este requisito mínimo, chamado de *score mínimo de correspondência*, é registrado na exibição do catálogo DB2GSE.ST_GEOCODING_PARAMETERS.

Tabela 35. Colunas na Exibição do Catálogo DB2GSE.ST_GEOCODING

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
TABLE_SCHEMA	VARCHAR(128)	Não	Nome do esquema que contém a tabela que contém a coluna identificada na coluna COLUMN_NAME.
TABLE_NAME	VARCHAR(128)	Não	Nome não qualificado da tabela que contém a coluna identificada na coluna COLUMN_NAME.
COLUMN_NAME	VARCHAR(128)	Não	Nome da coluna espacial a ser ocupada de acordo com as especificações mostradas nesta exibição do catálogo. Os valores combinados nas colunas TABLE_SCHEMA, TABLE_NAME e COLUMN_NAME identificam exclusivamente a coluna espacial.
GEOCODER_NAME	VARCHAR(128)	Não	Nome do geocoder que deve gerar dados para a coluna espacial especificada na coluna COLUMN_NAME. Somente um geocoder pode ser atribuído a uma coluna espacial.
MODE	VARCHAR(128)	Não	Modo para o processo de geocoding: **BATCH** Somente o geocoding em batch está ativado. **AUTO** O geocoding automático está configurado e ativado. **INVALID** Foi detectada uma inconsistência nas tabelas do catálogo espacial; a entrada de geocoding é inválida.
SOURCE_COLUMNS	VARCHAR(10000)	Sim	Nomes de colunas da tabela configuradas para geocoding automático. Sempre que estas colunas forem atualizadas, um disparo solicitará ao geocoder que efetue geocode dos dados atualizados.
WHERE_CLAUSE	VARCHAR(10000)	Sim	Condição de pesquisa em uma cláusula WHERE. Esta condição indica que quando o geocoder é executado no modo batch, ele efetua geocode somente dos dos em um subconjunto de registros especificado.
COMMIT_COUNT	INTEGER	Sim	O número de linhas que devem ser processadas durante o geocoding em batch antes da emissão de uma consolidação. Se o valor na coluna COMMIT_COUNT for 0 (zero) ou nulo, nenhuma consolidação será emitida.

A Exibição do Catálogo DB2GSE.ST_GEOCODING_PARAMETERS

Ao configurar as operações de geocoding para um determinado geocoder, os aspectos específicos do geocoder das definições são registrados automaticamente no catálogo do Spatial Extender. Por exemplo, uma operação específica do geocoder fornecido, DB2GSE_USA_GEOCODER, é comparar endereços fornecidos como entrada para dados de referência e para efetuar geocode na matriz, se eles corresponderem os mais recentes a um grau especificado ou a um grau maior do que o especificado. Ao configurar operações para este geocoder, especifique o que este grau, chamado de *score mínimo de correspondência*, deve ser; e sua especificação é registrada no catálogo.

Para saber os aspectos específicos do geocoder das definições para operações de geocoding, consulte a exibição do catálogo DB2GSE.ST_GEOCODING_PARAMETERS. Esta exibição é descrita na tabela a seguir.

Alguns padrões para configurações de operações de geocoding estão disponíveis na exibição do catálogo DB2GSE.ST_GEOCODER_PARAMETERS. Os valores na exibição DB2GSE.ST_GEOCODING_PARAMETERS substituem os padrões.

Tabela 36. Colunas na Exibição do Catálogo DB2GSE.ST_GEOCODING_PARAMETERS

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
TABLE_SCHEMA	VARCHAR(128)	Não	Nome do esquema que contém a tabela que contém a coluna identificada na coluna COLUMN_NAME.
TABLE_NAME	VARCHAR(128)	Não	Nome não qualificado da tabela que contém a coluna espacial.
COLUMN_NAME	VARCHAR(128)	Não	Nome da coluna espacial a ser ocupada de acordo com as especificações mostradas nesta exibição do catálogo. Os valores combinados nas colunas TABLE_SCHEMA, TABLE_NAME e COLUMN_NAME identificam exclusivamente esta coluna espacial.
ORDINAL	SMALLINT	Não	A posição deste parâmetro (ou seja, o parâmetro especificado na coluna PARAMETER_NAME) na assinatura da função que serve como o geocoder para a coluna identificada na coluna COLUMN_NAME. Um registro na exibição do catálogo SYSCAT.ROUTINEPARMS do DB2 também contém informações sobre este parâmetro. Este registro contém um valor que aparece na coluna ORDINAL de SYSCAT.ROUTINEPARMS. Este valor é igual ao que aparece na coluna ORDINAL da exibição DB2GSE.ST_GEOCODING_PARAMETERS.
PARAMETER_NAME	VARCHAR(128)	Sim	Nome de um parâmetro na definição do geocoder. Se nenhum nome foi especificado quando o geocoder foi definido, PARAMETER_NAME será nulo. Este conteúdo da coluna PARAMETER_NAME é obtido do catálogo do DB2.

Tabela 36. Colunas na Exibição do Catálogo DB2GSE.ST_GEOCODING_PARAMETERS (continuação)

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
PARAMETER_VALUE	VARCHAR(2048)	Sim	O valor que é atribuído a este parâmetro. O DB2 interpretará este valor como uma expressão SQL. Se o valor estiver entre aspas, ele será transmitido para o geocoder como uma cadeia. Caso contrário, a avaliação da expressão SQL determinará qual será o tipo de dados do parâmetro quando ele for transmitido para o geocoder. Se a coluna PARAMETER_VALUE contiver um nulo, este nulo será transmitido para o geocoder. A coluna PARAMETER_VALUE corresponde à coluna PARAMETER_DEFAULT na exibição do catálogo DB2GSE.ST_GEOCODER_PARAMETERS. Se a coluna PARAMETER_VALUE contiver um valor, este valor substituirá o valor padrão na coluna PARAMETER_DEFAULT. Se a coluna PARAMETER_VALUE for nula, será utilizado o valor padrão.

A Exibição do Catálogo DB2GSE.ST_SIZINGS

Utilize a exibição do catálogo DB2GSE.ST_SIZINGS para recuperar:

- Todas as variáveis suportadas pelo Spatial Extender; por exemplo, *nome do sistema de coordenadas*, *nome do geocoder* e variáveis às quais as representações de texto convencional de dados espaciais podem ser atribuídas.
- O comprimento máximo permitido, se conhecido, de valores atribuídos a estas variáveis (por exemplo, os comprimentos máximos permitidos de nomes de sistemas de coordenadas, de nomes de geocoders e de representações de texto convencional de dados espaciais).

Para obter uma descrição das colunas na exibição, consulte a tabela a seguir.

Tabela 37. Colunas na Exibição do Catálogo DB2GSE.ST_SIZINGS

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
VARIABLE_NAME	VARCHAR(128)	Não	Termo que indica uma variável. O termo é exclusivo no banco de dados.
SUPPORTED_VALUE	INTEGER	Sim	Comprimento máximo permitido dos valores atribuídos à variável mostrada na coluna VARIABLE_NAME. Os possíveis valores na coluna SUPPORTED_VALUE são: **Um valor numérico diferente de 0** O comprimento máximo permitido de valores atribuídos a esta variável. **0** Qualquer comprimento é permitido, ou o comprimento permitido não pode ser determinado. **NULL** O Spatial Extender não suporta esta variável.
DESCRIPTION	VARCHAR(128)	Sim	Descrição desta variável.

A Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS

Consulte a Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS para recuperar informações sobre sistemas de referência espacial registrados. O Spatial Extender registra automaticamente sistemas de referência espacial no catálogo do Spatial Extender nas seguintes situações:

- Quando você ativa um banco de dados para operações espaciais, cinco sistemas de referência espacial padrão e 318 sistemas de referência espacial geodésicos predefinidos. Para obter detalhes, consulte "Decidindo pela Utilização de um Sistema de Referência Espacial Padrão ou pela Criação de um Novo Sistema" na página 68 e "Datums Suportados pelo DB2 Geodetic Extender" na página 213.
- Quando os usuários criam sistemas de referência espacial adicionais.

Para obter o valor completo da exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, é necessário entender que cada sistema de referência espacial está associado a um sistema de coordenadas. O sistema de referência espacial é projetado parcialmente para converter coordenadas derivadas do sistema de coordenadas em valores que o DB2 pode processar com o máximo de eficiência e, parcialmente, para definir a máxima extensão possível de espaço que pode ser referido por essas coordenadas.

Para saber o nome e tipo do sistema de coordenadas associado a um determinado sistema de referência espacial, consulte as colunas COORDSYS_NAME e COORDSYS_TYPE da exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS. Para obter mais informações sobre o sistema de coordenadas, consulte a exibição do catálogo DB2GSE.ST_COORDINATE_SYSTEMS.

Tabela 38. Colunas na Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
SRS_NAME	VARCHAR(128)	Não	Nome do sistema de referência espacial. Este nome é exclusivo no banco de dados.
SRS_ID	INTEGER	Não	Identificador numérico do sistema de referência espacial. Cada sistema de referência espacial tem um identificador numérico exclusivo. Os sistemas de referência espacial geodésicos possuem valores SRS_ID no intervalo de 2000000000 a 2000001000. As funções espaciais especificam sistemas de referência espacial por seus identificadores numéricos em vez de especificar por seus nomes.
X_OFFSET	DOUBLE	Não	O deslocamento a ser subtraído de todas as coordenadas X de uma geometria. A subtração é uma etapa no processo de conversão de coordenadas de geometria em valores que o DB2 pode processar com eficiência máxima. Uma etapa seguinte é multiplicar a figura resultante da subtração pelo fator de escala mostrado na coluna X_SCALE.
X_SCALE	DOUBLE	Não	Fator de escala pelo qual é multiplicada a figura resultante quando um deslocamento é subtraído de uma coordenada X. Este fator é idêntico ao valor mostrado na coluna Y_SCALE.

Tabela 38. Colunas na Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS (continuação)

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
Y_OFFSET	DOUBLE	Não	O deslocamento a ser subtraído de todas as coordenadas Y de uma geometria. A subtração é uma etapa no processo de conversão de coordenadas de geometria em valores que o DB2 pode processar com eficiência máxima. Uma etapa seguinte é multiplicar a figura resultante da subtração pelo fator de escala mostrado na coluna Y_SCALE.
Y_SCALE	DOUBLE	Não	Fator de escala pelo qual é multiplicada a figura resultante quando um deslocamento é subtraído de uma coordenada Y. Este fator é idêntico ao valor mostrado na coluna X_SCALE.
Z_OFFSET	DOUBLE	Não	O deslocamento a ser subtraído de todas as coordenadas Z de uma geometria. A subtração é uma etapa no processo de conversão de coordenadas de geometria em valores que o DB2 pode processar com eficiência máxima. Uma etapa seguinte é multiplicar a figura resultante da subtração pelo fator de escala mostrado na coluna Z_SCALE.
Z_SCALE	DOUBLE	Não	Fator de escala pelo qual é multiplicada a figura resultante quando um deslocamento é subtraído de uma coordenada Z.
M_OFFSET	DOUBLE	Não	O deslocamento a ser subtraído de todas as medidas associadas a uma geometria. A subtração é uma etapa no processo de conversão das medidas em valores que o DB2 pode processar com eficiência máxima. Uma etapa seguinte é multiplicar a figura resultante da subtração pelo fator de escala mostrado na coluna M_SCALE.
M_SCALE	DOUBLE	Não	Fator de escala pelo qual é multiplicada a figura resultante quando um deslocamento é subtraído de uma medida.
MIN_X	DOUBLE	Não	Valor mínimo possível para coordenadas X nas geometrias às quais este sistema de referência espacial se aplica. Este valor é derivado dos valores nas colunas X_OFFSET e X_SCALE.
MAX_X	DOUBLE	Não	Valor máximo possível para coordenadas X nas geometrias às quais este sistema de referência espacial se aplica. Este valor é derivado dos valores nas colunas X_OFFSET e X_SCALE.
MIN_Y	DOUBLE	Não	Valor mínimo possível para coordenadas Y nas geometrias às quais este sistema de referência espacial se aplica. Este valor é derivado dos valores nas colunas Y_OFFSET e Y_SCALE.
MAX_Y	DOUBLE	Não	Valor máximo possível para coordenadas Y nas geometrias às quais este sistema de referência espacial se aplica. Este valor é derivado dos valores nas colunas Y_OFFSET e Y_SCALE.

Tabela 38. Colunas na Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS (continuação)

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
MIN_Z	DOUBLE	Não	Valor mínimo possível para coordenadas Z nas geometrias às quais este sistema de referência espacial se aplica. Este valor é derivado dos valores nas colunas Z_OFFSET e Z_SCALE.
MAX_Z	DOUBLE	Não	Valor máximo possível para coordenadas Z nas geometrias às quais este sistema de referência espacial se aplica. Este valor é derivado dos valores nas colunas Z_OFFSET e Z_SCALE.
MIN_M	DOUBLE	Não	Valor mínimo possível para as medidas que podem ser armazenadas com as geometrias às quais este sistema de referência espacial se aplica. Este valor é derivado dos valores nas colunas M_OFFSET e M_SCALE.
MAX_M	DOUBLE	Não	Valor máximo possível para as medidas que podem ser armazenadas com as geometrias às quais este sistema de referência espacial se aplica. Este valor é derivado dos valores nas colunas M_OFFSET e M_SCALE.
COORDSYS_NAME	VARCHAR(128)	Não	Nome de identificação do sistema de coordenadas no qual este sistema de referência espacial está baseado.
COORDSYS_TYPE	VARCHAR(128)	Não	Tipo do sistema de coordenadas no qual este sistema de referência espacial está baseado.
ORGANIZATION	VARCHAR(128)	Sim	Nome da organização (por exemplo, um órgão de padronizações) que definiu o sistema de coordenadas no qual este sistema de referência espacial está baseado. ORGANIZATION será nulo se ORGANIZATION_COORSYS_ID for nulo.
ORGANIZATION_ COORDSYS_ID	INTEGER	Sim	Nome da organização (por exemplo, um órgão de padronizações) que definiu o sistema de coordenadas no qual este sistema de referência espacial está baseado. ORGANIZATION_COORDSYS_ID será nulo se ORGANIZATION for nulo.
DEFINITION	VARCHAR(2048)	Não	Representação de texto convencional da definição do sistema de coordenadas.
DESCRIPTION	VARCHAR(256)	Sim	Descrição do sistema de referência espacial.

Conceitos Relacionados:

- "Sistemas de Referência Espacial" na página 67

Tarefas Relacionadas:

- "Criando um Sistema de Referência Espacial" na página 74

A Exibição do Catálogo DB2GSE.ST_UNITS_OF_MEASURE

Determinadas funções espaciais aceitam ou retornam valores que indicam uma distância específica. Em alguns casos, você pode escolher em qual unidade de medida a distância deve ser expressada. Por exemplo, ST_Distance retorna a distância mínima entre duas geometrias especificadas. Em uma ocasião, você pode requerer que ST_Distance retorne a distância em milhas; em outra, pode requerer uma distância expressa em metros. Para saber entre quais unidades de medida você pode escolher, consulte a exibição do catálogo DB2GSE.ST_UNITS_OF_MEASURE.

Tabela 39. Colunas na Exibição do Catálogo DB2GSE.ST_UNITS_OF_MEASURE

Nome	Tipo de Dados	Pode Ser Anulada?	Conteúdo
UNIT_NAME	VARCHAR(128)	Não	Nome da unidade de medida. Este nome é exclusivo no banco de dados.
UNIT_TYPE	VARCHAR(128)	Não	Tipo da unidade de medida. Os possíveis valores são: **LINEAR** A unidade de medida é linear. **ANGULAR** A unidade de medida é angular.
CONVERSION_FACTOR	DOUBLE	Não	Valor numérico utilizado para converter esta unidade de medida em sua unidade base. A unidade base para unidades lineares de medida é METER; a unidade base para unidades angulares de medida é RADIAN. A própria unidade base tem um fator de conversão de 1.0.
DESCRIPTION	VARCHAR(256)	Sim	Descrição da unidade de medida.

Capítulo 22. Funções Espaciais: Categorias e Utilizações

Este capítulo apresenta todas as funções espaciais, organizando-as por categoria.

Funções Espaciais

O DB2® Spatial Extender fornece funções que:

- Convertem geometrias de e para vários formatos de troca de dados. Essas funções são chamadas de *funções construtoras*.
- Comparam geometrias para limites, interseções e outras informações. Essas funções são chamadas de *funções de comparação*.
- Retornam informações sobre as propriedades das geometrias, como coordenadas e medidas nas geometrias, relações entre as geometrias, além de limite e outras informações.
- Geram novas geometrias a partir de geometrias existentes.
- Medem a distância mais curta entre os pontos de geometrias.
- Fornecem informações sobre os parâmetros de índices.
- Fornecem projeções e conversões entre diferentes sistemas de coordenadas.

Conceitos Relacionados:

- "Função que Retorna Informações sobre Distância" na página 336
- "Função que Retorna Informações sobre Índice" na página 337
- "Conversões entre Sistemas Coordenados" na página 337

Referência Relacionada:

- "Exemplos de Como as Funções Espaciais Operam" na página 123
- "Funções que Retornam Informações sobre Propriedades Geométricas" na página 322
- "Funções que Comparam Recursos Geográficos" na página 308
- "Funções que Geram Novas Geometrias de Geometrias Existentes" na página 329
- "Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados" na página 299

Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados

O DB2 Spatial Extender fornece funções espaciais que convertem geometrias de e para os seguintes formatos de troca de dados:

- Representação WKT (Well-Known Text)
- Representação WKB (Well-Known Binary)
- Representação de Formatos ESRI
- Representação GML (Geography Markup Language)

As funções para a criação de geometrias a partir desses são conhecidas como *funções construtoras*.

Conceitos Relacionados:

- "Visão Geral das Funções Construtoras" na página 300
- "Conversão em Representação WKT (Well-Known Text)" na página 304
- "Conversão em Representação WKB (Well-Known Binary)" na página 305
- "Conversão em Representação de Formato ESRI" na página 306
- "Conversão em Representação GML (Geography Markup Language)" na página 307

Referência Relacionada:

- "Grupos de Transformação" na página 513

Visão Geral das Funções Construtoras

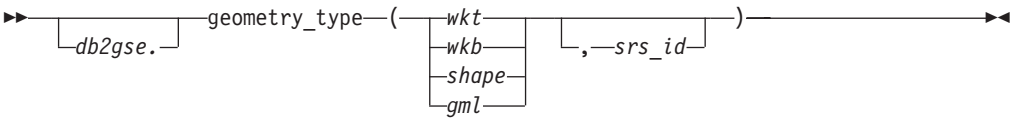
As funções construtoras possuem o mesmo nome do tipo de dados de geometria da coluna em que os dados serão inseridos. Essas funções operam de forma consistente em cada um dos formatos de troca de dados de entrada. Esta seção fornece:

- A SQL para chamar funções que operam em formatos de troca de dados e o tipo de geometria retornado por essas funções
- A SQL para chamar uma função que cria pontos a partir das coordenadas X e Y e o tipo de geometria retornado por essa função
- Exemplos de código e conjuntos de resultados

Funções que Operam em Formatos de Troca de Dados

Esta seção fornece a sintaxe para chamar funções que operam em formatos de troca de dados, descreve os parâmetros de entrada das funções e identifica o tipo de geometria retornado por essas funções.

Sintaxe:



Parâmetros e outros elementos da sintaxe:

db2gse Nome do esquema ao qual pertencem os tipos de dados espaciais fornecidos pelo DB2® Spatial Extender.

geometry_type
Uma das seguintes funções construtoras:

- ST_Point
- ST_LineString
- ST_Polygon
- ST_MultiPoint
- ST_MultiLineString
- ST_MultiPolygon
- ST_GeomCollection
- ST_Geometry

wkt Um valor do tipo CLOB(2 G) que contém a representação de texto reconhecido da geometria.

wkb Um valor do tipo BLOB(2 G) que contém a representação de binário reconhecido da geometria.

shape Um valor do tipo BLOB(2G) que contém a representação de formato ESRI da geometria.

gml Um valor do tipo CLOB(2 G) que contém a representação GML (Geography Markup Language) da geometria.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da geometria resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado.

Tipo de Retorno:

geometry_type

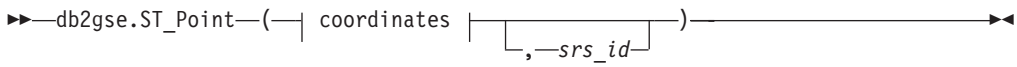
Se *geometry_type* for *ST_Geometry*, o tipo dinâmico do tipo de geometria retornado corresponderá à geometria indicada pelo valor de entrada.

Se *geometry_type* for de qualquer outro tipo, o tipo dinâmico do tipo de geometria retornado corresponderá ao nome da função. Se a geometria indicada pelo valor de entrada não corresponder ao nome da função ou ao nome de um de seus subtipos, será retornado um erro.

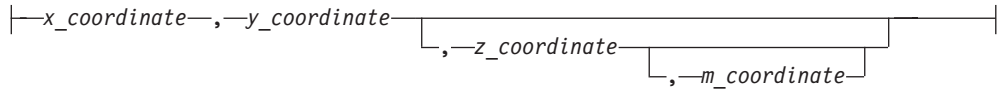
Uma Função que Cria Geometrias a Partir de Coordenadas

A função ST_Point cria geometrias não apenas de formatos de troca de dados, mas também de valores de coordenadas numéricas—uma capacidade muito útil se os dados de sua localização já estiverem armazenados em seu banco de dados. Esta seção fornece a sintaxe para chamar ST_Point, uma explicação de seus parâmetros e informações sobre o tipo de geometria retornado.

Sintaxe:



coordinates:



Parâmetros:

x_coordinate
Um valor do tipo DOUBLE que especifica a coordenada X do ponto resultante.

y_coordinate
Um valor do tipo DOUBLE que especifica a coordenada Y do ponto resultante.

z_coordinate
Um valor do tipo DOUBLE que especifica a coordenada Z do ponto resultante.

Se o parâmetro *z_coordinate* for omitido, o ponto resultante não terá uma coordenada Z. O resultado de ST_Is3D é 0 (zero) para esse ponto.

m_coordinate
Um valor do tipo DOUBLE que especifica a coordenada M do ponto resultante.

Se o parâmetro *m_coordinate* for omitido, o ponto resultante não terá uma medida. O resultado de ST_IsMeasured é 0 (zero) para esse ponto.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do ponto resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, surgirá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Point

Exemplos

Esta seção fornece exemplos de código para a chamada de funções construtoras, código para criação de tabelas que conterão a saída de funções construtoras, código para recuperação da saída e a própria saída.

O exemplo a seguir insere uma linha na tabela SAMPLE_GEOMETRY com ID 100 e um valor de ponto com uma coordenada X de 30, uma coordenada Y de 40 e no sistema de referência espacial 1 utilizando a representação de coordenadas e a representação de WKT (texto reconhecido). Em seguida, insere outra linha com ID 200 e um valor de cadeia de linhas com as coordenadas indicadas.

```
CREATE TABLE sample_geometry (id INT, geom db2gse.ST_Geometry);

INSERT INTO sample_geometry(id, geom)
       VALUES(100,db2gse.ST_Geometry('point(30 40)', 1));

INSERT INTO sample_geometry(id, geom)
       VALUES(200,db2gse.ST_Geometry('linestring(50 50, 100 100', 1));

SELECT id, TYPE_NAME(geom) FROM sample_geometry

ID      2
------  --------------------------------
   100  "ST_POINT"
   200  "ST_LINESTRING"
```

Se você souber que a coluna espacial pode conter apenas valores ST_Point, pode utilizar o exemplo a seguir, que insere dois pontos. A tentativa de inserir uma cadeia de linhas ou qualquer outro tipo que não seja um ponto resultará em um erro de SQL. A primeira inserção cria uma geometria de ponto a partir da representação de WKT (texto reconhecido). A segunda cria uma geometria de

ponto a partir de valores de coordenadas numéricas. Observe que esses valores de entrada também poderiam ser selecionados a partir de colunas de tabelas existentes.

```
CREATE TABLE sample_points (id INT, geom db2gse.ST_Point);

INSERT INTO sample_points(id, geom)
       VALUES(100,db2gse.ST_Point('point(30 40)', 1));

INSERT INTO sample_points(id, geom)
       VALUES(101,db2gse.ST_Point(50, 50, 1));

SELECT id, TYPE_NAME(geom) FROM sample_geometry

ID      2
------  ------------------------------
   100 "ST_POINT"
   101 "ST_POINT"
```

O exemplo a seguir utiliza SQL incorporada e assume que o aplicativo preenche as áreas de dados com os valores apropriados.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS CLOB(10000) wkt_buffer;
   SQL TYPE IS CLOB(10000) gml_buffer;
   SQL TYPE IS BLOB(10000) wkb_buffer;
   SQL TYPE IS BLOB(10000) shape_buffer;
EXEC SQL END DECLARE SECTION;

// * A lógica do aplicativo para ler os buffers vai aqui */

EXEC SQL INSERT INTO sample_geometry(id, geom)
         VALUES(:id, db2gse.ST_Geometry(:wkt_buffer,1));

EXEC SQL INSERT INTO sample_geometry(id, geom)
         VALUES:id, db2gse.ST_Geometry(:wkb_buffer,1));

EXEC SQL INSERT INTO sample_geometry(id, geom)
         VALUES(:id, db2gse.ST_Geometry(:gml_buffer,1));

EXEC SQL INSERT INTO sample_geometry(id, geom)
         VALUES(:id, db2gse.ST_Geometry(:shape_buffer,1));
```

A amostra de código Java™ a seguir utiliza JDBC para inserir geometrias de pontos utilizando os valores de coordenadas numéricas X, Y e utiliza a representação de WKT para especificar as geometrias.

```
String ins1 = "INSERT into sample_geometry (id, geom)
              VALUES(?, db2gse.ST_PointFromText(CAST( ?
              as VARCHAR(128)), 1))";
PreparedStatement pstmt = con.prepareStatement(ins1);
pstmt.setInt(1, 100);                    // valor do id
pstmt.setString(2, "point(32.4 50.7)");  // valor do wkt
int rc = pstmt.executeUpdate();

String ins2 = "INSERT into sample_geometry (id, geom)
              VALUES(?, db2gse.ST_Point(CAST( ? as double),
              CAST(? as double), 1))";
pstmt = con.prepareStatement(ins2);
pstmt.setInt(1, 200);         // valor do id
pstmt.setDouble(2, 40.3);     // lat
pstmt.setDouble(3, -72.5);    // long
rc = pstmt.executeUpdate();
```

Referência Relacionada:

• "Grupos de Transformação" na página 513

Conversão em Representação WKT (Well-Known Text)

As representações de texto são valores CLOB representando cadeias de caracteres ASCII. Elas permitem que as geometrias sejam trocadas na forma de texto ASCII.

A função **ST_AsText** converte um valor de geometria armazenado em uma tabela em uma cadeia WKT. O exemplo a seguir utiliza uma consulta simples de linha de comando para selecionar os valores que foram inseridos anteriormente na tabela SAMPLE_GEOMETRY.

```
SELECT id, VARCHAR(db2gse.ST_AsText(geom), 50) AS WKTGEOM
FROM sample_geometry;

ID   WKTGEOM
------  --------------------------------
   100   POINT ( 30.00000000 40.00000000)
   200   LINESTRING ( 50.00000000 50.00000000, 100.00000000 100.00000000)
```

O exemplo a seguir utiliza SQL incorporada para selecionar os valores que foram inseridos anteriormente na tabela SAMPLE_GEOMETRY.

```
 EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
 SQL TYPE IS CLOB(10000) wkt_buffer;
 short wkt_buffer_ind = -1;
EXEC SQL END DECLARE SECTION;

 EXEC SQL
 SELECT id, db2gse.ST_AsText(geom)
 INTO :id, :wkt_buffer :wkt_buffer_ind
   FROM sample_geometry
 WHERE id = 100;
```

Alternativamente, você pode utilizar o grupo de transformação ST_WellKnownText para converter geometrias em sua representação de texto reconhecido ao ligá-los. O código de amostra a seguir mostra como utilizar o grupo de transformação.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS CLOB(10000) wkt_buffer;
   short wkt_buffer_ind = -1;
EXEC SQL END DECLARE SECTION;

EXEC SQL
   SET CURRENT DEFAULT TRANSFORM GROUP = ST_WellKnownText;

EXEC SQL
SELECT id, geom
  INTO :id, :wkt_buffer :wkt_buffer_ind
   FROM sample_geometry
  WHERE id = 100;
```

Não são utilizadas funções espaciais na instrução SELECT para converter a geometria.

Além das funções explicadas nesta seção, o DB2® Spatial Extender fornece outras funções que também convertem geometrias de e para representações de texto reconhecido. O DB2 Spatial Extender fornece essas outras funções para atender a especificação "Recursos Simples para SQL" do OGC e a Parte 3: Padrão espacial de ISO SQL/MM. Estas funções são:

- **ST_WKTToSQL**
- **ST_GeomFromText**
- **ST_GeomCollFromTxt**
- **ST_PointFromText**
- **ST_LineFromText**
- **ST_PolyFromText**
- **ST_MPointFromText**
- **ST_MLineFromText**
- **ST_MPolyFromText**

Referência Relacionada:

• "Grupos de Transformação" na página 513

Conversão em Representação WKB (Well-Known Binary)

A representação de WKB consiste em estruturas de dados binários que devem ser valores BLOB. Os valores BLOB representam estruturas de dados binários que devem ser gerenciadas por um programa aplicativo escrito em uma linguagem de programação suportada pelo DB2® e para a qual o DB2 tenha uma ligação de linguagem.

A função **ST_AsBinary** converte um valor de geometria armazenado em uma tabela na representação de WKB (binário reconhecido), que pode ser buscada em uma variável BLOB no armazenamento do programa. O exemplo a seguir utiliza SQL incorporada para selecionar os valores que foram inseridos anteriormente na tabela SAMPLE_GEOMETRY.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS BLOB(10000) wkb_buffer;
   short wkb_buffer_ind = -1;
EXEC SQL END DECLARE SECTION;

EXEC SQL
   SELECT id, db2gse.ST_AsBinary(geom)
   INTO :id, :wkb_buffer :wkb_buffer_ind
   FROM sample_geometry
   WHERE id = 200;
```

Alternativamente, você pode utilizar o grupo de transformação ST_WellKnownBinary para converter geometrias em sua representação de binário reconhecido ao ligá-los. O código de amostra a seguir mostra como utilizar esse grupo de transformação.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS BLOB(10000) wkb_buffer;
   short wkb_buffer_ind = -1;
EXEC SQL END DECLARE SECTION;

EXEC SQL
   SET CURRENT DEFAULT TRANSFORM GROUP = ST_WellKnownBinary;

EXEC SQL
SELECT id, geom
   INTO :id, :wkb_buffer :wkb_buffer_ind
   FROM sample_geometry
   WHERE id = 200;
```

Não são utilizadas funções espaciais na instrução SELECT para converter a geometria.

Além das funções explicadas nesta seção, existem outras funções que também convertem geometrias de e para representações de binário reconhecido. O DB2 Spatial Extender fornece essas outras funções para atender a especificação "Recursos Simples para SQL" do OGC e a Parte 3: Padrão espacial de ISO SQL/MM. Estas funções são:

- **ST_WKBToSQL**
- **ST_GeomFromWKB**
- **ST_GeomCollFromWKB**
- **ST_PointFromWKB**
- **ST_LineFromWKB**
- **ST_PolyFromWKB**
- **ST_MPointFromWKB**
- **ST_MLineFromWKB**
- **ST_MPolyFromWKB**

Referência Relacionada:

- "Grupos de Transformação" na página 513

Conversão em Representação de Formato ESRI

A representação de Formato ESRI consiste em estruturas de dados binários que devem ser gerenciadas por um programa aplicativo escrito em uma linguagem suportada.

A função **ST_AsShape** converte um valor de geometria armazenado em uma tabela na representação de Formato ESRI, que pode ser buscada em uma variável BLOB no armazenamento do programa. O exemplo a seguir utiliza SQL incorporada para selecionar os valores que foram inseridos anteriormente na tabela SAMPLE_GEOMETRY.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id;
   SQL TYPE IS BLOB(10000) shape_buffer;
EXEC SQL END DECLARE SECTION;

EXEC SQL
   SELECT id, db2gse.ST_AsShape(geom)
   INTO :id, :shape_buffer
   FROM sample_geometry;
```

Alternativamente, você pode utilizar o grupo de transformação ST_Shape para converter geometrias implicitamente em sua representação de forma ao ligá-las. O código de amostra a seguir mostra como utilizar o grupo de transformação.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS BLOB(10000) shape_buffer;
   short shape_buffer_ind = -1;
EXEC SQL END DECLARE SECTION;

EXEC SQL
   SET CURRENT DEFAULT TRANSFORM GROUP = ST_Shape;
```

```
EXEC SQL
SELECT id, geom
   FROM sample_geometry
   WHERE id = 300;
```

Não são utilizadas funções espaciais na instrução SELECT para converter a geometria.

Referência Relacionada:

- "Grupos de Transformação" na página 513

Conversão em Representação GML (Geography Markup Language)

Representações GML (Geography Markup Language) são cadeias ASCII. Elas permitem que as geometrias sejam trocadas na forma de texto ASCII.

A função **ST_AsGML** converte um valor de geometria armazenado em uma tabela em uma cadeia de texto GML. O exemplo a seguir seleciona os valores que foram inseridos anteriormente na tabela SAMPLE_GEOMETRY. Os resultados mostrados no exemplo foram reformatados para facilitar a leitura. O espaçamento nos resultados pode variar de acordo com a exibição on-line.

```
SELECT id, VARCHAR(db2gse.ST_AsGML(geom), 500) AS GMLGEOM
FROM sample_geometry;

ID           GMLGEOM
-----------  ------------------------------------------------------------
        100 <gml:Point srsName="EPSG:4269">
            <gml:coord><gml:X>30</gml:X><gml:Y>40</gml:Y></gml:coord>
            </gml:Point>
        200 <gml:LineString srsName="EPSG:4269">
            <gml:coord><gml:X>50</gml:X><gml:Y>50</gml:Y></gml:coord>
            <gml:coord><gml:X>100</gml:X><gml:Y>100</gml:Y></gml:coord>
            </gml:LineString>
```

Alternativamente, você pode utilizar o grupo de transformação ST_GML para converter geometrias implicitamente em sua representação de HTML ao ligá-las.

```
SET CURRENT DEFAULT TRANSFORM GROUP = ST_GML

SELECT id, geom AS GMLGEOM
FROM sample_geometry;


ID           GMLGEOM
-----------  ------------------------------------------------------------
        100  <gml:Point srsName="EPSG:4269">
             <gml:coord><gml:X>30</gml:X><gml:Y>40</gml:Y></gml:coord>
             </gml:Point>
        200  <gml:LineString srsName="EPSG:4269">
             <gml:coord><gml:X>50</gml:X><gml:Y>50</gml:Y></gml:coord>
             <gml:coord><gml:X>100</gml:X><gml:Y>100</gml:Y></gml:coord>
             </gml:LineString>
```

Não são utilizadas funções espaciais na instrução SELECT para converter a geometria.

Referência Relacionada:

- "Grupos de Transformação" na página 513

Funções que Comparam Recursos Geográficos

Determinadas funções espaciais retornam informações sobre as formas com que os recursos geográficos relacionam-se ou comparam-se entre si. Outras funções espaciais retornam informações sobre se duas definições de sistemas de coordenadas ou dois sistemas de referência espacial são os mesmos. Em todos os casos, as informações retornadas são resultados de uma comparação entre geometrias, entre definições de sistemas de coordenadas ou entre sistemas de referência espacial. As funções que fornecem essas informações são chamadas *funções de comparação.*

As funções de comparação são:

- **ST_Contains e ST_Within.** Essas funções obtêm duas geometrias como entrada e determinam se o interior de uma cruza o interior da outra.
- **ST_Intersects, ST_Crosses, ST_Overlaps e ST_Touches.** Essas funções retornam informações sobre interseções de geometrias.
- **ST_EnvIntersects e ST_MBRIntersects.** Essas funções determinam se o menor retângulo que inclui uma geometria cruza o menor retângulo que inclui outra geometria.
- **ST_Equals, ST_EqualCoordsys e ST_EqualSRS.** Essas funções determinam se duas coisas sendo comparadas são idênticas.
- **ST_Relate.** Essa função determina se as geometrias sendo comparadas atendem as condições da cadeia de matrizes padrão DE-9IM.
- **ST_Disjoint.** Essa função verifica a não interseção entre duas geometrias.

Conceitos Relacionados:

- "Função que Compara Geometrias à Cadeia de Matrizes Padrão DE-9IM" na página 322
- "Visão Geral das Funções de Comparação" na página 308
- "Funções que Verificam Se uma Geometria Contém Outra" na página 310
- "Funções que Verificam Interseções Entre as Geometrias" na página 313
- "Funções que Comparam Envelopes das Geometrias" na página 318
- "Funções que Verificam Se Duas Coisas São Idênticas" na página 319
- "Função que Verifica a não Interseção Entre Duas Geometrias" na página 321

Visão Geral das Funções de Comparação

As funções de comparação do DB2® Spatial Extender retornarão um valor 1 (um) se uma comparação atender determinados critérios, um valor 0 (zero) se uma comparação não atender os critérios e um valor nulo se a comparação não puder ser executada. As comparações não poderão ser executadas se a operação de comparação não tiver sido definida para os parâmetros de entrada, ou se um dos parâmetros for nulo. As comparações *poderão* ser executadas se as geometrias com tipos de dados ou dimensões diferentes forem atribuídas aos parâmetros.

O *DE-9IM (Dimensionally Extended 9 Intersection Model)* é uma abordagem matemática que define a relação espacial de par inteligente entre as geometrias de tipos e dimensões diferentes. Esse modelo expressa relações espaciais entre todos os tipos de geometrias como interseções de pares inteligentes de seu interior, limite e exterior, considerando-se a dimensão das interseções resultantes.

As geometrias especificadas *a* e *b*: I(*a*), B(*a*) e E(*a*) representam o interior, o limite e o exterior de *a*, respectivamente. E, I(*b*), B(*b*), e E(*b*) representam o interior, o limite, e o exterior de *b*. As interseções de I(*a*), B(*a*), e E(*a*) com I(*b*), B(*b*), e E(*b*) produzem uma matriz de 3–por–3. Cada interseção pode resultar em geometrias de dimensões diferentes. Por exemplo, a interseção dos limites de dois polígonos consiste em um ponto e uma cadeia de linhas, em cujo caso a função dim retorna a dimensão máxima de 1.

A função dim retorna um valor –1, 0, 1 ou 2. O valor –1 corresponde ao conjunto nulo ou dim(null), que é retornado quando nenhuma interseção é localizada.

	Interior	**Limite**	**Exterior**
Interior	dim(I(a) ∩ I(b))	dim(I(a) ∩ B(b))	dim(I(a) ∩ E(b))
Limite	dim(B(a) ∩ I(b))	dim(B(a) ∩ B(b))	dim(B(a) ∩ E(b))
Exterior	dim(E(a) ∩ I(b))	dim(E(a) ∩ B(b))	dim(E(a) ∩ E(b))

Os resultados retornados pelas funções de comparação podem ser entendidos ou verificados pela comparação dos resultados retornados por uma função de comparação com uma matriz padrão que representa os valores aceitáveis para o DE-9IM.

A matriz padrão contém os valores aceitáveis para cada uma das células matrizes de interseção. Os possíveis valores padrão são:

T Deve existir uma interseção; dim = 0, 1 ou 2.

F Não deve existir uma interseção; dim = -1.

***** Não importa se existir uma interseção; dim = -1, 0, 1 ou 2.

0 Deve existir uma interseção e sua dimensão exata deve ser 0; dim = 0.

1 Deve existir uma interseção e sua dimensão máxima deve ser 1; dim = 1.

2 Deve existir uma interseção e sua dimensão máxima deve ser 2; dim = 2.

Por exemplo, a matriz padrão a seguir para a função ST_Within inclui os valores T, F e *.

Tabela 40. Matriz de ST_Within. A matriz padrão da função ST_Within para combinações de geometrias.

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Geometry a Interior	T	*	F
Limite da Geometria a	*	*	F
Exterior da Geometria a	*	*	*

A função ST_Within retorna um valor 1 quando os interiores das duas geometrias se cruzam e quando o interior ou o limite de *a* não cruza o exterior de *b.* As outras condições não importam.

Cada função tem, pelo menos, uma matriz padrão, mas algumas requerem mais de uma para descrever as relações de várias combinações de tipos de geometria.

O DE-9IM foi desenvolvido por Clementini e Felice, que ampliaram dimensionalmente o Modelo de Interseção 9 de Egenhofer e Herring. O DE-9IM é

uma colaboração de quatro autores (Clementini, Eliseo, Di Felice e van Osstrom) que publicaram o modelo em "A Small Set of Formal Topological Relationships Suitable for End-User Interaction," D. Abel e B.C. Ooi (Ed.), *Advances in Spatial Database — no Terceiro Simpósio Internacional. SSD '93*. LNCS 692. Pp. 277-295. O modelo de interseção 9 desenvolvido por M. J. Egenhofer e J. Herring (Springer-Verlag Singapore [1993]) foi publicado em "Categorizing binary topological relationships between regions, lines, and points in geographic databases," *Tech. Report, Department of Surveying Engineering*, University of Maine, Orono, ME 1991.

Lista de funções

As funções de comparação são:

- ST_Contains
- ST_Crosses
- ST_Disjoint
- ST_EnvIntersects
- ST_EqualCoordsys
- ST_Equals
- ST_EqualSRS
- ST_Intersects
- ST_MBRIntersects
- ST_Overlaps
- ST_Relate
- ST_Touches
- ST_Within

Funções que Verificam Se uma Geometria Contém Outra

ST_Contains e ST_Within obtêm duas geometrias como entrada e determinam se o interior de uma cruza o interior da outra. Em termos coloquiais, ST_Contains determina se a primeira geometria especificada para ela inclui a segunda geometria (se a primeira contém a segunda). ST_Within determina se a primeira geometria está completamente dentro da segunda (se a primeira está contida na segunda).

ST_Contains

ST_Contains retornará um valor 1 (um) se a segunda geometria estiver completamente contida pela primeira geometria. A função ST_Contains retorna o resultado oposto exato da função ST_Within.

A Figura 44 na página 311 mostra exemplos de ST_Contains:

- Uma geometria multiponto contém geometrias de um ponto ou multiponto quando todos os pontos estão dentro da primeira geometria.
- Uma geometria de polígono contém uma geometria multiponto quando todos os pontos estão no limite do polígono ou no interior do polígono.
- Uma geometria de cadeia de linhas contém geometrias de um ponto, multiponto ou de cadeia de linhas quando todos os pontos estão dentro da primeira geometria.
- Uma geometria de polígono contém geometrias de um ponto ou de cadeia de linhas ou de polígono quando a segunda geometria está no interior do polígono.

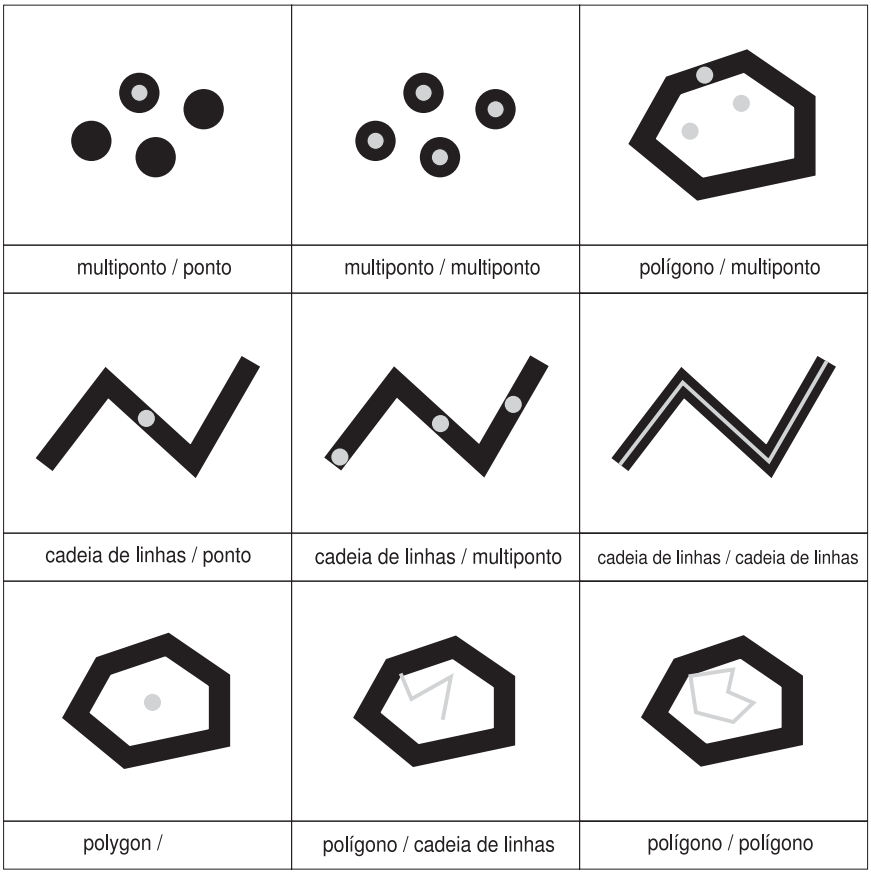


Figura 44. ST_Contains. As geometrias escuras representam a geometria ″a″ e as geometrias cinzas representam a geometria ″b″. Em todos os casos, a geometria *a* contém a geometria *b* completamente.

A matriz padrão da função ST_Contains indica que os interiores das duas geometrias devem se cruzar e que o interior ou limite da segunda (geometria *b*) não deve cruzar o exterior da principal (geometria *a*). O asterisco (*) indica que não importa se existe uma interseção entre estas partes das geometrias.

Tabela 41. Matriz de ST_Contains

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Interior da Geometria a	T	*	*
Limite da Geometria a	*	*	*
Exterior da Geometria a	F	F	*

ST_Within

ST_Within retornará um valor 1 (um) se a primeira geometria estiver completamente contida na segunda geometria. ST_Within retorna o resultado oposto exato de ST_Contains.

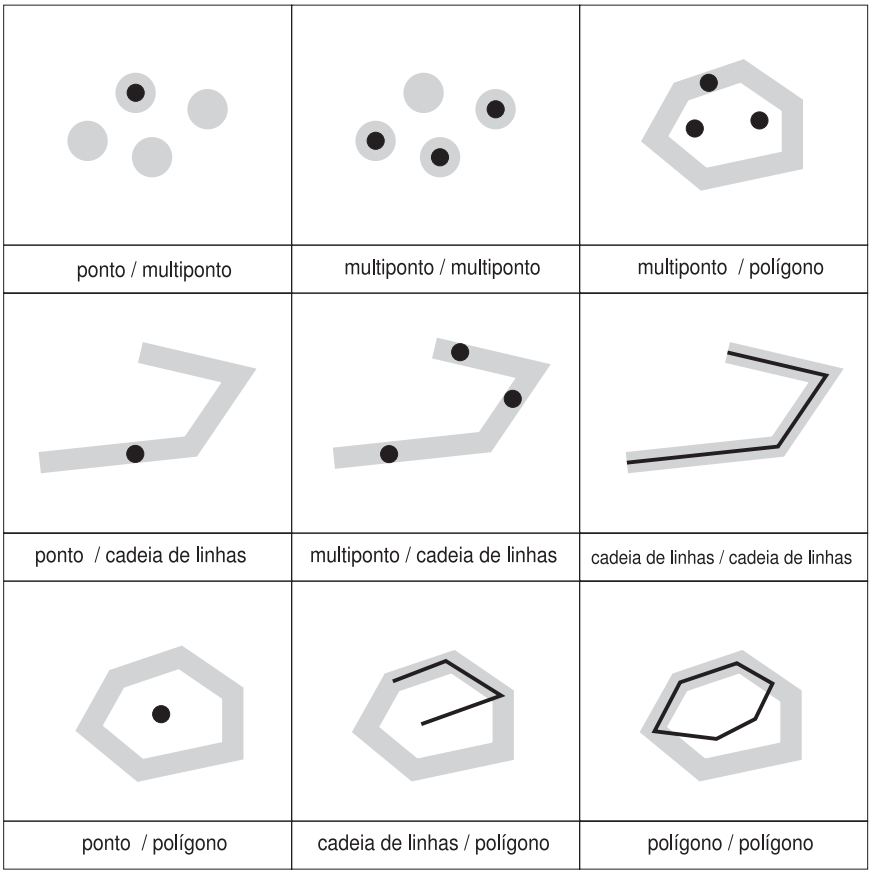


Figura 45. ST_Within

A matriz padrão da função ST_Within indica que os interiores das duas geometrias devem se cruzar e que o interior ou limite da geometria principal (geometria *a*) não deve cruzar o exterior da secundária (geometria *b*). O asterisco (*) indica que as demais interseções não importam.

Tabela 42. Matriz de ST_Within

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Interior da Geometria a	T	*	F
Limite da Geometria a	*	*	F
Exterior da Geometria a	*	*	*

A Figura 45 mostra exemplos de ST_Within:

- Uma geometria de ponto está dentro de uma geometria multiponto quando seu interior cruza um dos pontos na segunda geometria.
- Uma geometria multiponto está dentro de uma geometria multiponto quando os interiores de todos os pontos cruzam a segunda geometria.
- Uma geometria multiponto está dentro de uma geometria de polígono quando todos os pontos estão no limite do polígono ou no interior do polígono.
- Uma geometria de ponto está dentro de uma geometria de cadeia de linhas quando todos os pontos estão dentro da segunda geometria. Na Figura 45, o ponto não está dentro da cadeia de linhas porque seu interior não cruza a cadeia

de linhas; no entanto, a geometria multiponto está dentro da cadeia de linhas porque todos os seus pontos cruzam o interior da cadeia de linhas.
- Uma geometria de cadeia de linhas está dentro de outras geometrias de cadeia de linhas quando todos os seus pontos cruzam a segunda geometria.
- Uma geometria de ponto não está dentro de uma geometria de polígono porque seu interior não cruza o limite ou o interior do polígono.
- Uma geometria de cadeia de linhas está dentro de uma geometria de polígono quando todos os seus pontos cruzam o limite ou o interior do polígono.
- Uma geometria de polígono está dentro de uma geometria de polígono quando todos os seus pontos cruzam o limite ou o interior do polígono.

Funções que Verificam Interseções Entre as Geometrias

ST_Intersects, ST_Crosses, ST_Overlaps e ST_Touches determinam se uma geometria cruza com outra. A principal diferença entre elas é o escopo de interseção para o qual são utilizadas para testar:

- ST_Intersects testa para determinar se as duas geometrias especificadas para ela atendem uma das quatro condições: a de interseção dos interiores das geometrias, a de interseção de seus limites, a do limite da interseção da primeira geometria com o interior da segunda ou a do interior de interseção da primeira geometria com o limite da segunda.
- ST_Crosses é utilizada para analisar a interseção de geometrias de dimensões diferentes, com uma exceção: também pode analisar a interseção de cadeias de linhas. Em todos os casos, o próprio local de interseção é considerado uma geometria; e ST_Crosses requer que essa geometria tenha uma dimensão menor do que a maior das interseções de geometrias (ou, se ambas forem cadeias de linhas, que o local de interseção tenha uma dimensão menor do que uma cadeia de linhas). Por exemplo, as dimensões de uma cadeia de linhas e um polígono são 1 e 2, respectivamente. Se essas duas geometrias se cruzarem, e se o local de interseção for linear (o caminho da cadeia de linhas junto ao polígono), esse próprio local poderá ser considerado uma cadeia de linhas. E como a dimensão de uma cadeia de linhas (1) é menor do que a de um polígono (2), ST_Crosses, depois de analisar a interseção, retornaria um valor 1.
- As geometrias especificadas para ST_Overlaps como entrada devem ter a mesma dimensão. ST_Overlaps requer que essas geometrias sobreponham-se parcialmente, formando uma nova geometria (a região de sobreposição) que tem a mesma dimensão delas.
- ST_Touches determina se os limites das duas geometrias se cruzam.

ST_Intersects

ST_Intersects retornará um valor 1 (um) se a interseção não resultar em um conjunto vazio. ST_Intersects retorna o resultado oposto exato de ST_Disjoint.

A função ST_Intersects retornará 1 (um) se as condições de qualquer uma das matrizes padrão a seguir retornar TRUE.

Tabela 43. Matriz de ST_Intersects (1). A função ST_Intersects retornará 1 (um) se os interiores de ambas as geometrias se cruzarem.

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Limite da Geometria a	*	*	*
Interior da Geometria a	T	*	*
Exterior da Geometria a	*	*	*

Tabela 44. Matriz de ST_Intersects (2). A função ST_Intersects retornará 1 (um) se o limite da primeira geometria cruzar o limite da segunda geometria.

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Limite da Geometria a	*	*	*
Interior da Geometria a	*	T	*
Exterior da Geometria a	*	*	*

Tabela 45. Matriz de ST_Intersects (3). A função ST_Intersects retornará 1 (um) se o limite da primeira geometria cruzar o interior da segunda.

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Limite da Geometria a	T	*	*
Interior da Geometria a	*	*	*
Exterior da Geometria a	*	*	*

Tabela 46. Matriz de ST_Intersects (4). A função ST_Intersects retornará 1 (um) se os limites de qualquer das geometrias se cruzarem.

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Limite da Geometria a	*	T	*
Interior da Geometria a	*	*	*
Exterior da Geometria a	*	*	*

ST_Crosses

ST_Crosses utiliza duas geometrias e retornará um valor 1 (um) se:

- A interseção resultar em uma geometria cuja dimensão seja menor do que a dimensão máxima das geometrias de origem.
- O conjunto de interseções for interior a ambas as geometrias de origem.

ST_Crosses retornará um valor nulo se a primeira geometria for uma superfície ou multisuperfície ou se a segunda geometria for um ponto ou multiponto. Para as outras combinações, ST_Crosses retornará um valor 1 (indicando que as duas geometrias se cruzam) ou um valor 0 (indicando que não se cruzam).

A figura a seguir ilustra multipontos que cruzam a cadeia de linhas, a cadeia de linhas que cruza a cadeia de linhas, vários pontos que cruzam um polígono e cadeia de linhas que cruza um polígono. Em três dos quatro casos, a geometria *b* cruza a geometria *a*. No quarto caso, a geometria a é um multiponto que não cruza a linha, mas toca a área dentro do polígono da geometria b.

As geometrias escuras representam a geometria *a*; e as geometrias cinza representam a geometria *b*.

Figura 46. ST_Crosses

A matriz padrão em Tabela 47 será aplicada se a primeira geometria for um ponto ou multiponto, ou se a primeira geometria for uma curva ou multicurva e a segunda geometria for uma superfície. A matriz indica que os interiores devem se cruzar e que o interior da principal (geometria *a*) deve cruzar o exterior da secundária (geometria *b*).

Tabela 47. Matriz para ST_Crosses (1)

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Limite da Geometria a	*	*	*
Interior da Geometria a	T	*	T
Exterior da Geometria a	*	*	*

A matriz padrão em Tabela 48 na página 316 será aplicada se a primeira e segunda geometrias forem ambas curvas ou multicurvas. O 0 indica que a interseção dos interiores deve ser um ponto (dimensão 0). Se a dimensão dessa interseção for 1 (cruza em uma cadeia de linhas), a função ST_Crosses retornará um valor 0 (indicando que as geometrias não se cruzam); entretanto, a função ST_Overlaps retornará um valor 1 (indicando que as geometrias se sobrepõem).

Tabela 48. Matriz para ST_Crosses (2)

	Interior da Geometria b	Limite da Geometria b	Exterior da Geometria b
Limite da Geometria a	*	*	*
Interior da Geometria a	0	*	*
Exterior da Geometria a	*	*	*

ST_Overlaps

ST_Overlaps compara duas geometrias da mesma dimensão. Retornará um valor 1 (um) se seu conjunto de interseções resultar em uma geometria diferente de ambas, mas com a mesma dimensão.

As geometrias escuras representam a geometria *a*; as geometrias cinzas representam a geometria *b*. Em todos os casos, as duas geometrias possuem a mesma dimensão e uma sobrepõe a outra parcialmente. A área de sobreposição é uma nova geometria; tem a mesma dimensão das geometrias *a* e *b*.

A figura a seguir ilustra sobreposições em geometrias. Os três exemplos mostram sobreposições com pontos, cadeias de linhas e polígonos. Com pontos, os pontos reais são sobrepostos. Com cadeias de linhas, uma parte da linha é sobreposta. Com polígonos, uma parte da área é sobreposta.

Figura 47. ST_Overlaps

A matriz padrão em Tabela 49 será aplicada se a primeira e segunda geometrias forem ambas pontos, multipontos, superfícies ou multisuperfícies. ST_Overlaps retornará um valor 1 se o interior de cada geometria cruzar o interior e exterior da outra geometria.

Tabela 49. Matriz para ST_Overlaps (1)

	Interior da Geometria b	Limite da Geometria b	Exterior da Geometria b
Limite da Geometria a	*	*	*
Interior da Geometria a	T	*	T
Exterior da Geometria a	T	*	*

A matriz padrão em Tabela 50 na página 317 será aplicada se a primeira e segunda geometrias forem ambas curvas ou multicurvas. Nesse caso, a interseção das geometrias deve resultar em uma geometria que tem uma dimensão de 1 (outra

curva). Se a dimensão da interseção dos interiores for 0, ST_Overlaps retornará um valor 0 (indicando que as geometrias não se sobrepõem); entretanto, a função ST_Crosses retornaria um valor 1 (indicando que as geometrias se cruzam).

Tabela 50. Matriz para ST_Overlaps (2)

	Interior da Geometria b	Limite da Geometria b	Exterior da Geometria b
Limite da Geometria a	*	*	*
Interior da Geometria a	1	*	T
Exterior da Geometria a	T	*	*

ST_Touches

ST_Touches retornará um valor 1 (um) se todos os pontos comuns a ambas as geometrias puderem ser localizados apenas nos limites. Os interiores das geometrias não devem se cruzar. Pelo menos uma geometria deve ser uma curva, superfície, multicurva ou multisuperfície.

As geometrias escuras representam a geometria *a*; as geometrias cinzas representam a geometria *b*. Em todos os casos, o limite da geometria *b* cruza a geometria *a*. O interior da geometria *b* permanece separado da geometria *a*.

A figura a seguir mostra exemplos de toque com tipos de geometrias, como um ponto e cadeia de linhas, cadeia de linhas e cadeia de linhas, ponto e polígono, multiponto e polígono e cadeia de linhas e polígono.

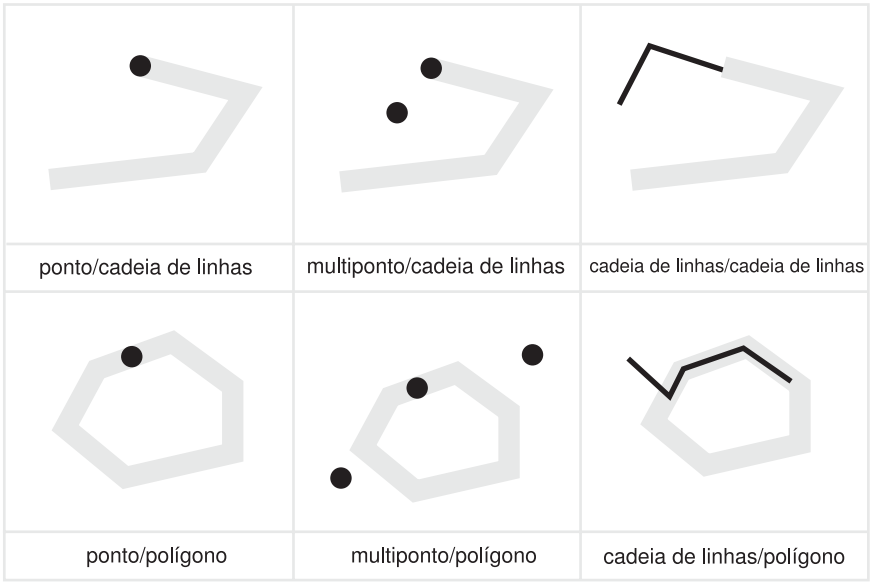


Figura 48. ST_Touches

As matrizes padrão mostram que a função ST_Touches retorna 1 (um) quando os interiores da geometria não se cruzam, e o limite de uma das geometrias cruza o interior da outra ou seu limite.

Tabela 51. Matriz de ST_Touches (1)

	Interior da Geometria b	Limite da Geometria b	Exterior da Geometria b
Limite da Geometria a	*	*	*
Interior da Geometria a	F	T	*
Exterior da Geometria a	*	*	*

Tabela 52. Matriz de ST_Touches (2)

	Interior da Geometria b	Limite da Geometria b	Exterior da Geometria b
Limite da Geometria a	T	*	*
Interior da Geometria a	F	*	*
Exterior da Geometria a	*	*	*

Tabela 53. Matriz de ST_Touches (3)

	Interior da Geometria b	Limite da Geometria b	Exterior da Geometria b
Limite da Geometria a	*	T	*
Interior da Geometria a	F	*	*
Exterior da Geometria a	*	*	*

Funções que Comparam Envelopes das Geometrias

ST_EnvIntersects e ST_MBRIntersects são semelhantes por determinarem se o menor retângulo que inclui uma geometria cruza o menor retângulo que inclui outra geometria. Tradicionalmente, esse retângulo é chamado de *envelope*. Multipolígonos, polígonos, cadeias de múltiplas linhas e cadeias de linhas curvas tocam os lados de seus envelopes; cadeias de linhas horizontais, cadeias de linhas verticais e pontos são levemente menores que seus envelopes. ST_EnvIntersects testa para determinar se os envelopes de geometrias se cruzam.

A menor área retangular na qual uma geometria pode se ajustar é chamada de MBR (Minimum Bounding Rectangle). Os envelopes em volta dos multipolígonos, polígonos, cadeias de múltiplas linhas e cadeias de linhas curvas são realmente MBRs. Mas os envelopes em volta de cadeias de linhas horizontais, cadeias de linhas verticais e pontos não são MBRs, porque não constituem uma área mínima em que essas últimas geometrias se encaixam. Essas últimas geometrias não ocupam espaço definível e, portanto, não podem ter MBRs. Contudo, foi adotada uma convenção por meio da qual são referidas como seus próprios MBRs. Portanto, em relação aos multipolígonos, polígonos, cadeias de múltiplas linhas e cadeias de linhas curvas, ST_MBRIntersects testa a interseção dos mesmos retângulos circundantes que ST_EnvIntersects testa. Mas para cadeias de linhas horizontais, cadeias de linhas verticais e pontos, ST_MBRIntersects testa as interseções dessas próprias geometrias.

ST_EnvIntersects

ST_EnvIntersects retornará um valor 1 (um) se os envelopes de duas geometrias se cruzarem. É uma função conveniente que implementa com eficiência ST_Intersects (`ST_Envelope(g1),ST_Envelope(g2)`).

ST_MBRIntersects

ST_MBRIntersects retornará um valor 1 (um) se os MBRs de duas geometrias se cruzarem.

Funções que Verificam Se Duas Coisas São Idênticas

ST_EqualCoordsys

ST_EqualCoordsys retorna um valor 1 (um) se duas definições do sistema de coordenadas forem idênticas. Ao comparar as definições, ST_EqualCoordsys desconsidera as diferenças de tipo de letra, espaços, parênteses e representação de números de pontos flutuantes.

ST_Equals

ST_Equals retornará um valor 1 (um) se duas geometrias forem idênticas. A ordem dos pontos utilizados para definir as geometrias não é relevante para o teste de igualdade.

Nos seis exemplos (ponto, multiponto, cadeia de linhas, multicadeia, polígono e multipolígono), a geometria a e a geometria b são iguais.

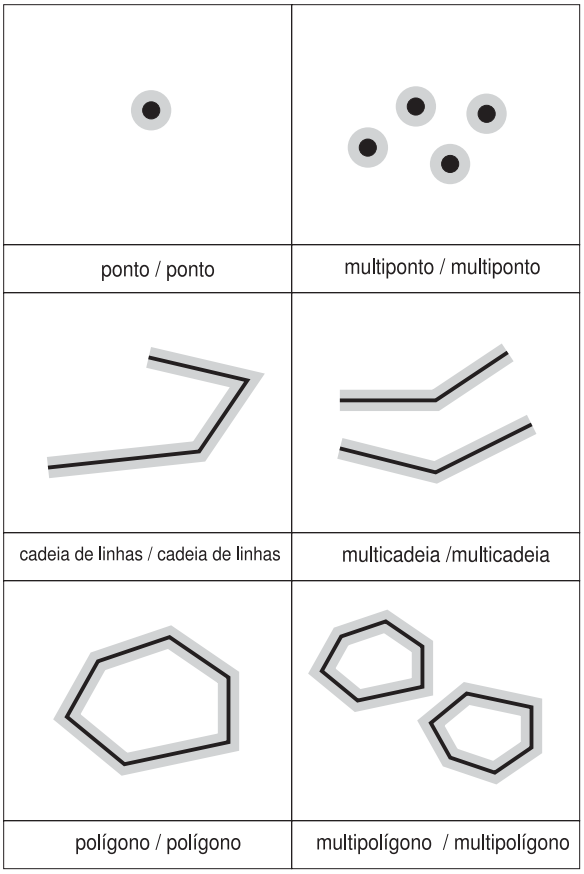


Figura 49. ST_Equals. As geometrias escuras representam a geometria *a*; as geometrias cinzas representam a geometria *b*. Em todos os casos, a geometria *a* é igual à geometria *b*.

Tabela 54. Matriz da Igualdade. A matriz padrão DE-9IM da igualdade assegura que os interiores se cruzam e que nenhuma parte interior ou limite das geometrias cruza o exterior da outra.

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Limite da Geometria a	*	*	F
Interior da Geometria a	T	*	F
Exterior da Geometria a	F	F	*

ST_EqualSRS

ST_EqualSRS retornará um valor 1 (um) se dois sistemas de referência espacial forem idênticos, contanto que o identificador numérico de um ou de ambos os sistemas não seja nulo.

Função que Verifica a não Interseção Entre Duas Geometrias

ST_Disjoint retornará um valor 1 (um) se a interseção das duas geometrias for um conjunto vazio. Essa função retorna o oposto exato do que é retornado por ST_Intersects.

A ilustração mostra diferentes geometrias e como os limites não se cruzam em nenhum ponto.

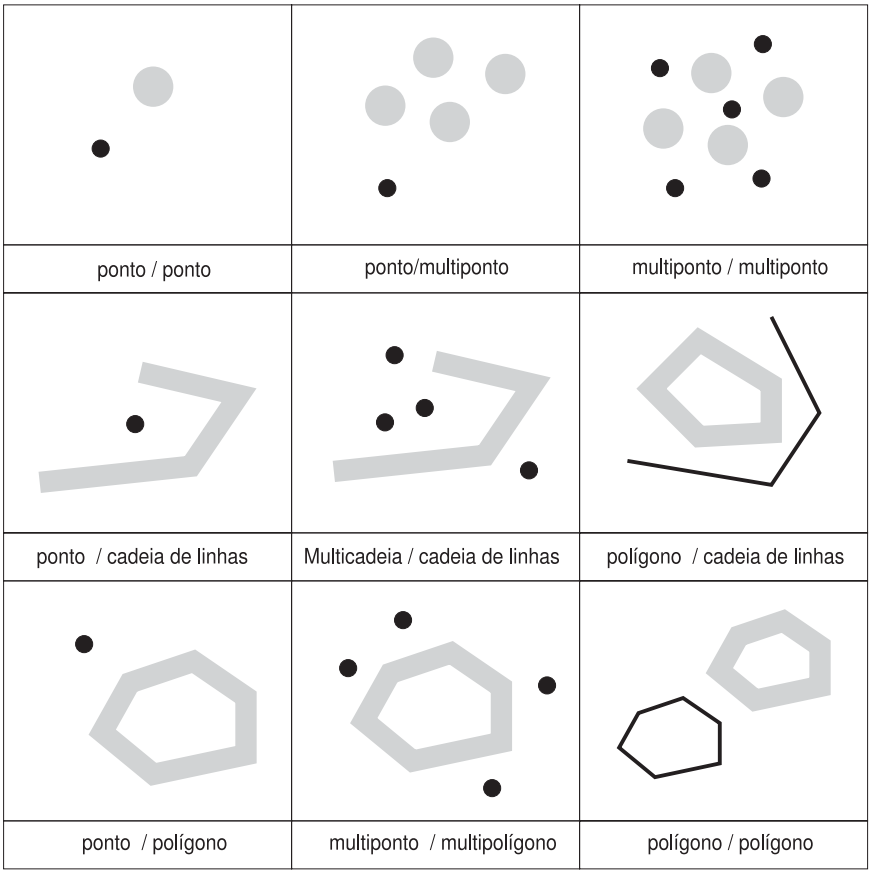


Figura 50. ST_Disjoint. As geometrias escuras representam a geometria *a*; as geometrias cinzas representam a geometria *b*. Em todos os casos, a geometria *a* e a geometria *b* são desconectadas uma da outra.

Tabela 55. Matriz para ST_Disjoint. Essa matriz apenas indica que nenhum dos interiores ou limites das geometrias se cruzam.

	Interior da Geometria b	**Limite da Geometria b**	**Exterior da Geometria b**
Limite da Geometria a	F	F	*
Interior da Geometria a	F	F	*
Exterior da Geometria a	*	*	*

Função que Compara Geometrias à Cadeia de Matrizes Padrão DE-9IM

A função ST_Relate compara duas geometrias e retorna um valor 1 (um) se as geometrias atenderem as condições especificadas pela cadeia de matrizes padrão DE-9IM; caso contrário, retorna um valor 0 (zero).

Funções que Retornam Informações sobre Propriedades Geométricas

Esta seção apresenta as funções espaciais que retornam informações sobre propriedades geométricas. Essas informações referem-se a:

- Tipos de dados de geometrias
- Coordenadas e medidas em uma geometria
- Anéis, limites, envelopes e MBRs (Minimum Bounding Rectangles)
- Dimensões
- As qualidades de ser fechado, vazio ou simples
- Geometrias base em uma coleção de geometrias
- Sistemas de referência espacial

Algumas propriedades são geometrias em seu próprio direito; por exemplo, os anéis exteriores e interiores de uma superfície, ou os pontos iniciais e nós de extremidade de uma curva. Essas geometrias são produzidas por algumas das funções nessa categoria. Funções que produzem outros tipos de geometrias—por exemplo, geometrias que representam zonas que envolvem uma determinada localização—pertencem a outra categoria. Para obter informações sobre essa outra categoria, chamada "Funções espaciais que geram novas geometrias", consulte o link apropriado ou a referência cruzada no final dessa seção.

Conceitos Relacionados:

- "Função que Retorna Informações sobre Tipo de Dados" na página 322
- "Funções que Retornam Informações sobre Coordenadas e Medidas" na página 323
- "Funções que Retornam Informações sobre Geometrias Contidas em uma Geometria" na página 325
- "Funções que Mostram Informações sobre Limites, Envelopes e Anéis" na página 326
- "Funções que Retornam Informações sobre as Dimensões de uma Geometria" na página 327
- "Funções que Revelam se uma Geometria É Fechada, Vazia ou Simples" na página 328
- "Funções que Identificam o Sistema de Referência Espacial de uma Geometria" na página 328

Função que Retorna Informações sobre Tipo de Dados

ST_GeometryType utiliza uma geometria como parâmetro de entrada e retorna o nome completo do tipo dinâmico dessa geometria.

Funções que Retornam Informações sobre Coordenadas e Medidas

As seguintes funções retornam informações sobre as coordenadas e medidas contidas em uma geometria. Por exemplo, ST_X pode retornar a coordenada X contida em um ponto especificado, ST_MaxX retorna a maior coordenada X contida em uma geometria e ST_MinX retorna a menor coordenada X contida em uma geometria.

Estas funções são:

- ST_CoordDim
- ST_IsMeasured
- ST_IsValid
- ST_Is3D
- ST_M
- ST_MaxM
- ST_MaxX
- ST_MaxY
- ST_MaxZ
- ST_MinM
- ST_MinX
- ST_MinY
- ST_MinZ
- ST_X
- ST_Y
- ST_Z

ST_CoordDim

ST_CoordDim retorna um valor que indica os tipos de coordenadas contidos em uma geometria e se a geometria também contém alguma medida. Esse valor chama-se *dimensão da coordenada*. Uma dimensão de coordenada não é o mesmo que a propriedade referida como *dimensão*. A última indica se uma geometria tem largura ou comprimento, não se contém coordenadas de um tipo específico ou medidas.

ST_IsMeasured

ST_IsMeasured utiliza uma geometria como um parâmetro de entrada e retorna 1 se a geometria especificada tiver coordenadas M (medidas). Caso contrário, será retornado 0 (zero).

ST_IsValid

ST_IsValid utiliza uma geometria como parâmetro de entrada e retornará 1 se for válida. Caso contrário, será retornado 0 (zero). Uma geometria será válida apenas se todos os atributos no tipo estruturado forem consistentes com a representação interna de dados da geometria e se a representação interna não estiver corrompida.

ST_Is3D

ST_Is3d utiliza uma geometria como um parâmetro de entrada e retorna 1 se a geometria especificada tiver coordenadas Z. Caso contrário, será retornado 0 (zero).

ST_M

Se uma medida for armazenada com um determinado ponto, ST_M poderá utilizar o ponto como parâmetro de entrada e retornar a medida.

ST_MaxM

ST_MaxM utiliza uma geometria como parâmetro de entrada e retorna sua medida máxima.

ST_MaxX

ST_MaxX utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada X máxima.

ST_MaxY

ST_MaxY utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada Y máxima.

ST_MaxZ

ST_MaxZ utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada Z máxima.

ST_MinM

ST_MinM utiliza uma geometria como parâmetro de entrada e retorna sua medida mínima.

ST_MinX

ST_MinX utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada X mínima.

ST_MinY

ST_MinY utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada Y mínima.

ST_MinZ

ST_MinY utiliza uma geometria como parâmetro de entrada e retorna sua coordenada Z mínima.

ST_X

ST_X pode utilizar um ponto como parâmetro de entrada e retornar a coordenada X do ponto.

ST_Y

ST_Y pode utilizar um ponto como parâmetro de entrada e retornar a coordenada Y do ponto.

ST_Z

Se uma coordenada Z for armazenada com um determinado ponto, ST_Z poderá utilizar o ponto como parâmetro de entrada e retornar a coordenada Z.

Funções que Retornam Informações sobre Geometrias Contidas em uma Geometria

As seguintes funções retornam informações sobre geometrias contidas em uma geometria. Algumas funções identificam pontos específicos contidos em uma geometria; outras retornam o número de geometrias base contidas em uma coleção.

Estas funções são:

- ST_Centroid
- ST_EndPoint
- ST_GeometryN
- ST_LineStringN
- ST_MidPoint
- ST_NumGeometries
- ST_NumLineStrings
- ST_NumPoints
- ST_NumPolygons
- ST_PointN
- ST_PolygonN
- ST_StartPoint

ST_Centroid

ST_Centroid utiliza uma geometria como um parâmetro de entrada e retorna o centro geométrico, que é o centro do retângulo limitador mínimo da geometria especificada, como um ponto.

ST_EndPoint

ST_Endpoint utiliza uma curva como um parâmetro de entrada e retorna o ponto que é o último ponto da curva.

ST_GeometryN

ST_GeometryN utiliza uma coleção de geometria e um índice como parâmetros de entrada e retorna a geometria na coleção que é identificada pelo índice.

ST_LineStringN

ST_LineStringN utiliza uma cadeia de múltiplas linhas e um índice como parâmetros de entrada e retorna a cadeia de linhas que é identificada pelo índice.

ST_MidPoint

ST_MidPoint utiliza uma curva como um parâmetro de entrada e retorna o ponto na curva que é eqüidistante de ambos os pontos de extremidade da curva, medido ao longo da curva.

ST_NumGeometries

ST_NumGeometries utiliza uma coleção de geometrias como parâmetro de entrada e retorna o número de geometrias da coleção.

ST_NumLineStrings

ST_NumLineStrings utiliza uma cadeia de múltiplas linhas como parâmetro de entrada e retorna o número de cadeias de linhas contido.

ST_NumPoints

ST_NumPoints utiliza uma geometria como parâmetro de entrada e retorna o número de pontos que foram utilizados para definir essa geometria. Por exemplo, se a geometria for um polígono e foram utilizados cinco pontos para definir esse polígono, o número retornado será 5.

ST_NumPolygons

ST_NumPolygons utiliza um multipolígono como parâmetro de entrada e retorna o número de polígonos contidos.

ST_PointN

ST_PointN utiliza uma cadeia de linhas ou um multiponto e um índice como parâmetros de entrada e retorna esse ponto na cadeia de linhas ou um multiponto que é identificado pelo índice.

ST_PolygonN

ST_PolygonN utiliza um multipolígono e um índice como parâmetros de entrada e retorna o polígono que é identificado pelo índice.

ST_StartPoint

ST_StartPoint utiliza uma curva como parâmetro de entrada e retorna o ponto que é o primeiro da curva.

Funções que Mostram Informações sobre Limites, Envelopes e Anéis

As funções a seguir retornam informações sobre demarcações que dividem uma parte interna de uma geometria a partir de uma parte externa, ou que divide a própria geometria a partir do espaço externo a ela. Por exemplo, ST_Boundary retorna o limite de uma geometria na forma de uma curva.

Estas funções são:

- ST_Boundary
- ST_Envelope
- ST_EnvIntersects
- ST_ExteriorRing
- ST_InteriorRingN
- ST_MBR
- ST_MBRIntersects
- ST_NumInteriorRing
- ST_Perimeter

ST_Boundary

ST_Boundary utiliza uma geometria como um parâmetro de entrada e retorna seu limite como uma nova geometria.

ST_Envelope

ST_Envelope utiliza uma geometria como um parâmetro de entrada e retorna um envelope em torno da geometria. O envelope é um retângulo que é representado como um polígono.

ST_EnvIntersects

ST_EnvIntersects utiliza duas geometrias como parâmetros de entrada e retorna 1 se os envelopes de duas geometrias forem interseccionados. Caso contrário, será retornado 0 (zero).

ST_ExteriorRing

ST_ExteriorRing utiliza um polígono como um parâmetro de entrada e retorna seu anel externo como uma curva.

ST_InteriorRingN

ST_InteriorRingN utiliza um polígono e um índice como parâmetros de entrada e retorna o anel interno identificado pelo índice especificado como uma cadeia de linhas. Os anéis internos são organizados de acordo com as regras definidas pelas rotinas de verificação de geometria interna.

ST_MBR

ST_MBR utiliza uma geometria como um parâmetro de entrada e retorna seu retângulo limitador mínimo.

ST_MBRIntersects

ST_MBRIntersects retornará um valor 1 (um) se os MBRs de duas geometrias se cruzarem.

ST_NumInteriorRing

ST_NumInteriorRing utiliza um polígono como parâmetro de entrada e retorna o número de seus anéis interiores.

ST_Perimeter

ST_Perimeter utiliza uma superfície ou multisuperfície e, opcionalmente, uma unidade como parâmetros de entrada e retorna o perímetro da superfície ou da multisuperfície (isto é, o comprimento de seu limite) medido nas unidades especificadas.

Funções que Retornam Informações sobre as Dimensões de uma Geometria

As funções a seguir retornam informações sobre a dimensão de uma geometria. Por exemplo, ST_Area informa a quantidade de área coberta por uma determinada geometria.

Estas funções são:

- ST_Area
- ST_Dimension
- ST_Length

ST_Area

ST_Area utiliza uma geometria e, opcionalmente, uma unidade como parâmetros de entrada e retorna a área coberta pela geometria especificada na unidade de medida especificada.

ST_Dimension

ST_Dimension utiliza uma geometria como um parâmetro de entrada e retorna sua dimensão.

ST_Length

ST_Length utiliza uma curva ou multicurva e, opcionalmente, uma unidade como parâmetros de entrada e retorna o comprimento da curva ou multicurva especificada na unidade de medida especificada.

Funções que Revelam se uma Geometria É Fechada, Vazia ou Simples

As seguintes funções indicam:

- Se uma determinada curva ou multicurva é fechada (isto é, se o ponto inicial e final da curva ou multicurva são os mesmos)
- Se uma determinada geometria é vazia (isto é, sem pontos)
- Se uma curva, multicurva ou multiponto é simples (isto é, se essas geometrias possuem configurações típicas)

ST_IsClosed

ST_IsClosed utiliza uma curva ou multicurva como parâmetro de entrada e retornará 1 se a curva ou multicurva especificada for fechada. Caso contrário, será retornado 0 (zero).

ST_IsEmpty

ST_IsEmpty utiliza uma geometria como um parâmetro de entrada e retorna 1 se a geometria especificada for vazia. Caso contrário, será retornado 0 (zero).

ST_IsSimple

ST_IsSimple utiliza uma geometria como um parâmetro de entrada e retorna 1 se a geometria especificada for simples. Caso contrário, será retornado 0 (zero).

Funções que Identificam o Sistema de Referência Espacial de uma Geometria

As seguintes funções retornam valores que identificam o sistema de referência espacial que foi associado à geometria. Além disso, a função ST_SrsID pode alterar o sistema de referência espacial da geometria sem alterar ou transformar a geometria.

ST_SrsId (Também Chamada ST_SRID)

ST_SrsId (ou ST_SRID) utiliza uma geometria e, opcionalmente, um identificador de sistema de referência espacial como parâmetros de entrada. O que é retornado por essa função depende dos parâmetros de entrada especificados:

- Se o identificador de sistema de referência espacial for especificado, a função retornará a geometria com seu sistema de referência espacial alterado para o sistema de referência espacial especificado. A transformação da geometria não é executada.
- Se nenhum identificador de sistema de referência espacial for especificado como parâmetro de entrada, o identificador de sistema de referência espacial atual da geometria especificada será retornado.

ST_SrsName

ST_SrsName utiliza uma geometria como um parâmetro de entrada e retorna o nome do sistema de referência espacial no qual a geometria especificada é representada.

Funções que Geram Novas Geometrias de Geometrias Existentes

Esta seção apresenta a categoria de funções que originam novas geometrias de geometrias existentes. Essa categoria não inclui funções que originam geometrias que representam propriedades de outras geometrias. Em vez disso, serve para funções que:

- Convertem geometrias em outras geometrias
- Criam geometrias que representam configurações de espaço
- Originam geometrias individuais de várias geometrias
- Criam geometrias com base em medidas
- Criam modificações de geometrias

Conceitos Relacionados:

- "Funções que Convertem uma Geometria em Outra" na página 329
- "Funções que Criam Novas Geometrias com Configurações de Espaço Diferentes" na página 330
- "Funções que Originam uma Geometria de Várias" na página 334
- "Funções que Originam Novas Geometrias com Base em Medidas" na página 334
- "Funções que Criam Formas Modificadas de Geometrias Existentes" na página 335

Funções que Convertem uma Geometria em Outra

As funções a seguir podem converter geometrias de um supertipo em geometrias correspondentes de um subtipo. Por exemplo, a função ST_ToLineString pode converter uma cadeia de linhas do tipo ST_Geometry em uma cadeia de linhas de ST_LineString. Algumas dessas funções também podem combinar geometrias base e coleções de geometrias em uma única coleção de geometrias. Por exemplo, ST_ToMultiLine pode converter uma cadeia de linhas e uma cadeia de múltiplas linhas em uma única cadeia de múltiplas linhas.

ST_Polygon

ST_Polygon pode construir um polígono a partir de uma cadeia de linhas fechada. A cadeia de linhas definirá o anel exterior do polígono.

ST_ToGeomColl

ST_ToGeomColl utiliza uma geometria como um parâmetro de entrada e a converte em uma coleção de geometria.

ST_ToLineString

ST_ToLineString utiliza uma geometria como um parâmetro de entrada e a converte em uma cadeia de linhas.

ST_ToMultiLine

ST_ToMultiLine utiliza uma geometria como um parâmetro de entrada e a converte em uma cadeia de múltiplas linhas.

ST_ToMultiPoint

ST_ToMultiPoint utiliza uma geometria como um parâmetro de entrada e a converte em um multiponto.

ST_ToMultiPolygon

ST_ToMultiPolygon utiliza uma geometria como um parâmetro de entrada e a converte em um multipolígono.

ST_ToPoint

ST_ToPoint utiliza uma geometria como um parâmetro de entrada e a converte em um ponto.

ST_ToPolygon

ST_ToPolygon utiliza uma geometria como um parâmetro de entrada e a converte em um polígono.

Funções que Criam Novas Geometrias com Configurações de Espaço Diferentes

Utilizando geometrias existentes como ponto inicial, as funções a seguir criam novas geometrias que representam áreas circulares ou outras configurações de espaço. Por exemplo, fornecido um ponto que representa o centro de um aeroporto indicado, ST_Buffer pode criar uma superfície que representa, na forma circular, a extensão proposta do aeroporto.

Estas funções são:

- ST_Buffer
- ST_ConvexHull
- ST_Difference
- ST_Intersection
- ST_SymDifference

ST_Buffer

A função ST_Buffer pode gerar uma nova geometria que se estende para fora de uma geometria existente por um raio especificado. A nova geometria será uma superfície quando a geometria existente for colocada em buffer ou sempre que os elementos de uma coleção estiverem tão próximos que os buffers em volta dos

únicos elementos da coleção serão sobrepostos. No entanto, quando os buffers forem separados, resultarão superfícies de buffers individuais, nesse caso ST_Buffer retornará uma multisuperfície.

A figura a seguir ilustra o buffer em torno de elementos únicos e sobrepostos.

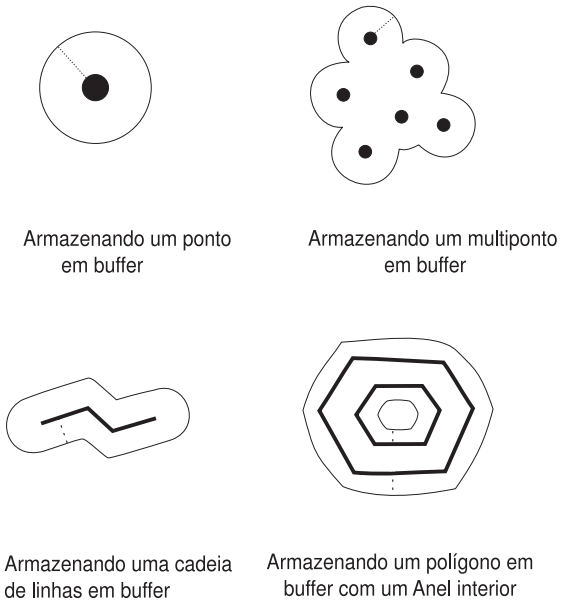


Figura 51. ST_Buffer

A função ST_Buffer aceita as distâncias positiva e negativa; entretanto, apenas as geometrias com uma dimensão de dois (superfícies e multisuperfícies) aplicam um buffer negativo. O valor absoluto da distância do buffer será utilizado sempre que a dimensão da geometria de origem for menor que 2 (todas as geometrias que não forem superfícies ou multisuperfícies).

Em geral, para anéis exteriores, as distâncias de buffer positivo geram anéis de superfície que estão fora do centro da geometria de origem; as distâncias de buffer negativo geram anéis de superfície ou multisuperfície voltados para o centro. Para anéis interiores de uma superfície ou multisuperfície, uma distância de buffer positivo gera um anel de buffer voltado para o centro e uma distância de buffer negativo gera um anel de buffer fora do centro.

O processo de colocação em buffer mescla superfícies que se sobrepõem. Distâncias negativas maiores que a metade da largura interior máxima de um polígono resulta em uma geometria vazia.

ST_ConvexHull

A função ST_ConvexHull retorna a cobertura exterior convexa de qualquer geometria que tenha pelo menos três vértices formando um convexo. *Vértices* são os pares de coordenadas X e Y nas geometrias. Uma *cobertura exterior convexa* é o menor polígono convexo que pode ser formado por todos os vértices em um determinado conjunto de vértices.

A ilustração a seguir mostra quatro exemplos de invólucros convexos. No primeiro exemplo, um formato irregular que lembra que a letra c foi desenhada. O c está fechado pelo invólucro convexo. No quarto exemplo, existem quatro pontos com

linhas em um padrão de zigue-zague. A linha convexa vai entre os pontos quatro e dois de um lado e três e um do outro lado.

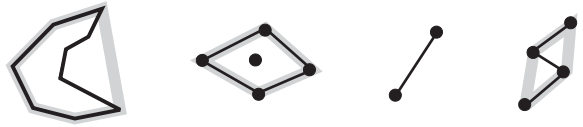


Figura 52. ST_ConvexHull

ST_Difference

ST_Difference utiliza duas geometrias da mesma dimensão como entrada. A função ST_Difference retorna essa parte da primeira geometria que não é cruzada pela segunda geometria. Essa operação é o equivalente espacial do AND NOT lógico. A parte da geometria retornada por ST_Difference é por si só uma geometria—uma coleção que possui a mesma dimensão das geometrias utilizadas como entrada. Se essas duas geometrias forem iguais—isto é, se ocuparem o mesmo espaço— a geometria retornada será vazia.

À esquerda de cada seta estão duas geometrias que são especificadas para ST_Difference como entrada. À direita de cada seta está a saída de ST_Difference. Se parte da primeira geometria for cruzada pela segunda, a saída será a parte da primeira geometria que não foi cruzada. Se as geometrias especificadas como entrada forem iguais, a saída será uma geometria vazia (indicada pelo termo *nil*)

Esta figura ilustra a entrada e saída para ST_Difference. Por exemplo, se a entrada for pontos, e um ponto a e um ponto b forem iguais, a saída será nula. Se um ponto a e um ponto b forem diferentes, a saída será um novo ponto entre os dois. Se a entrada for um polígono para b e um polígono menor mas idêntico para a geometria a dentro do primeiro, o resultado será nulo. Se os polígonos forem polígonos de sobreposição, a saída será as bordas externas dos polígonos combinados.

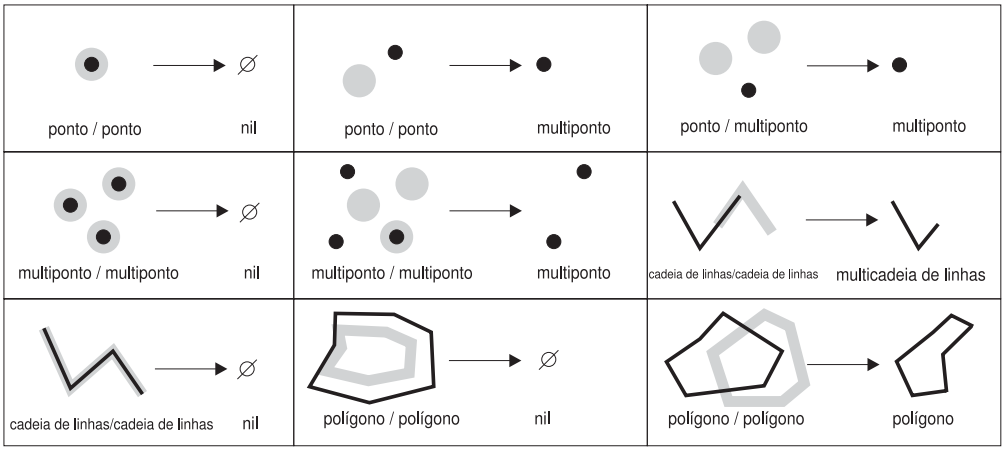


Figura 53. ST_Difference

ST_Intersection

A função ST_Intersection retorna um conjunto de pontos, representado como uma geometria, que define a interseção de duas geometrias especificadas. Se as geometrias especificadas para ST_Intersection como entrada não se cruzarem, ou se

cruzarem e a dimensão de sua interseção for menor do que as dimensões das geometrias, ST_Intersection retornará uma geometria vazia.

À esquerda de cada seta existem duas geometrias em interseção que são especificadas para ST_Intersection como entrada. À direita de cada seta está a saída de ST_Intersection—uma geometria que representa a interseção criada pelas geometrias à esquerda.

Esta figura ilustra dez exemplos de saída de ST_Intersection, que retorna informações sobre onde as geometrias fornecidas se cruzam. Por exemplo, se b for uma cadeia de linhas e a geometria a for um ponto na linha, a saída será o multiponto no qual a geometria a e a geometria b convergem. Se a geometria a e a geometria b forem polígonos de sobreposição, a saída será um novo multipolígono apenas da parte que foi sobreposta.

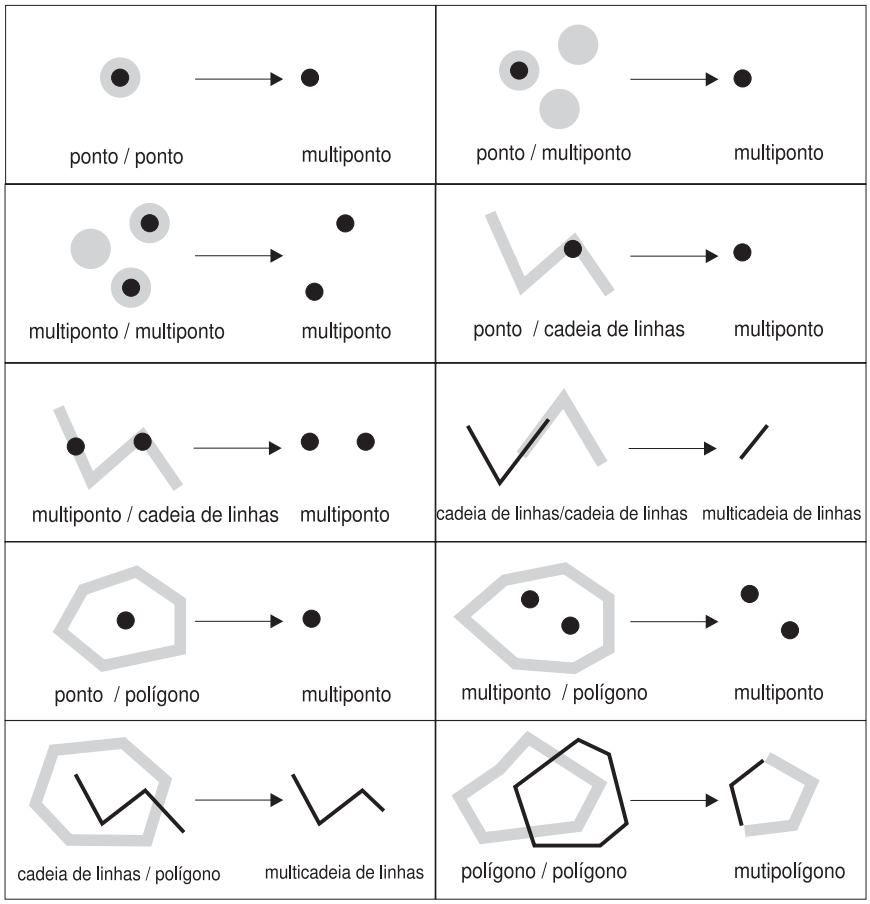


Figura 54. ST_Intersection

ST_SymDifference

A função ST_SymDifference retorna a diferença simétrica (o equivalente espacial da operação lógica XOR) de duas geometrias em interseção que possuem a mesma dimensão. Se essas geometrias forem iguais, ST_SymDifference retornará uma geometria vazia. Se não forem iguais, uma parte de uma ou de ambas ficará fora da área de interseção.

Funções que Originam uma Geometria de Várias

As funções a seguir originam geometrias individuais de várias geometrias. Por exemplo, ST_Union combina duas geometrias em uma única geometria.

Agregado de MBR

A combinação das funções ST_BuildMBRAggr e ST_GetAggrResult agrega uma coluna de geometrias em uma coluna selecionada em uma única geometria, construindo um retângulo que representa o retângulo de limite mínimo que inclui todas as geometrias na coluna. As coordenadas Z e M são descartadas quando o agregado é calculado.

ST_Union

A função ST_Union retorna o conjunto de união de duas geometrias. Essa operação é o equivalente espacial do OR lógico. As duas geometrias devem ter a mesma dimensão. ST_Union sempre retorna o resultado como uma coleção.

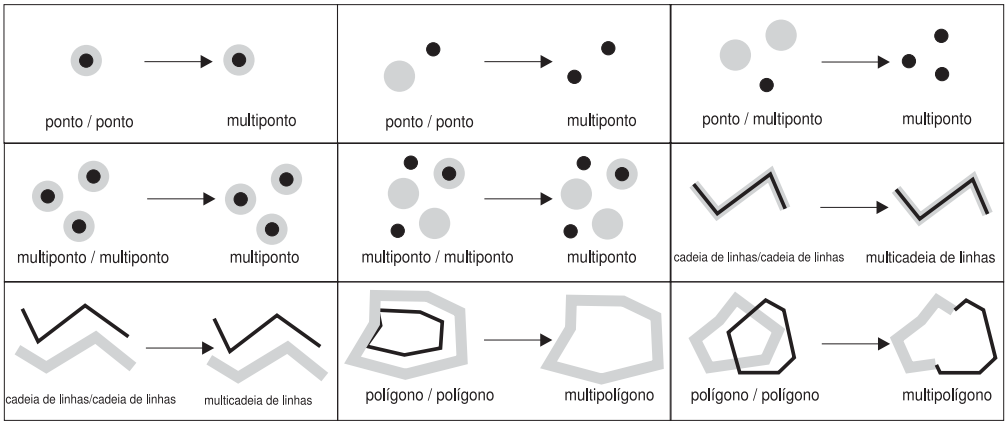


Figura 55. ST_Union

Agregado de União

Um agregado de união é a combinação das funções ST_BuildUnionAggr e ST_GetAggrResult. Essa combinação agrega uma coluna de geometrias em uma tabela em uma única geometria pela construção da união.

Funções que Originam Novas Geometrias com Base em Medidas

As funções explicadas nesta seção podem criar geometrias cujos pontos são associados a uma medida específica ou a uma seqüência específica de duas medidas. Por exemplo, suponha que as medidas que variam entre os valores 4 e 8 sejam armazenadas com os pontos em uma multicurva. Se quiser saber com quais pontos uma medida com um valor 7 é armazenada, poderá utilizar a função ST_FindMeasure para retornar esses pontos em um único multiponto.

Estas funções são:

- ST_FindMeasure (Também Chamada ST_LocateAlong)
- ST_MeasureBetween (Também Chamada ST_LocateBetween)

ST_FindMeasure (Também Chamada ST_LocateAlong)

ST_FindMeasure (ou ST_LocateAlong) utiliza uma geometria e uma medida como parâmetros de entrada. Retorna um multiponto ou multicurva da geometria especificada que correspondeu à medida especificada. Para pontos e multipontos, todos os pontos com a medida especificada são retornados. Para curvas, multicurvas, superfícies e multisuperfícies, a interpolação é executada para calcular o resultado. O cálculo para superfícies e multisuperfícies é executado no limite da geometria.

ST_MeasureBetween (Também Chamada ST_LocateBetween)

ST_MeasureBetween (ou ST_LocateBetween) utiliza uma geometria e duas coordenadas (medidas) M como parâmetros de entrada e retorna essa parte da geometria especificada que representa o conjunto de caminhos ou pontos desconectados entre as duas coordenadas M.

Para curvas, multicurvas, superfícies e multisuperfícies, a interpolação é executada para calcular o resultado. Na Figura 56 os pontos 3, 4, 5, 6, 7, 8 e 9 representam uma curva. Se as duas coordenadas M forem 4 e 7, ST_MeasureBetween retornará a parte da curva entre os pontos 4 e 7.

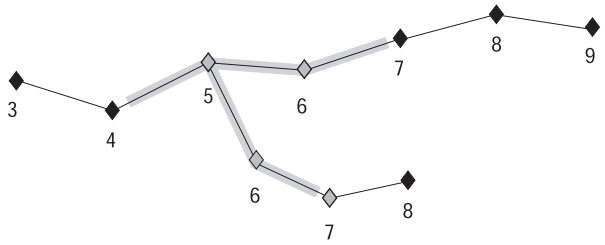


Figura 56. LocateBetween

Funções que Criam Formas Modificadas de Geometrias Existentes

As funções a seguir criam formas modificadas de geometrias existentes. Por exemplo, a função ST_AppendPoint cria versões estendidas de curvas existentes. Cada versão inclui os pontos em uma curva existente mais um ponto adicional.

Estas funções são:

- ST_AppendPoint
- ST_ChangePoint
- ST_Generalize
- ST_M
- ST_PerpPoints
- ST_RemovePoint
- ST_X
- ST_Y
- ST_Z

ST_AppendPoint

ST_AppendPoint utiliza uma curva e um ponto como parâmetros de entrada e estende a curva pelo ponto especificado.

ST_ChangePoint

ST_ChangePoint utiliza uma curva e dois pontos como parâmetros de entrada. Ela substitui todas as ocorrências do primeiro ponto na curva especificada pelo segundo ponto e retorna a curva resultante.

ST_Generalize

ST_Generalize utiliza uma geometria e um limite como parâmetros de entrada e representa a geometria especificada com um número reduzido de pontos, ao mesmo tempo que preserva as características gerais da geometria. O algoritmo de simplificação de linha Douglas-Peucker é utilizado, através do qual a seqüência de pontos que definem a geometria é recursivamente subdivida até que uma sucessão dos pontos possa ser substituída por um segmento linear reto. Neste segmento de linha, nenhum dos pontos de definição é desviado do segmento de linha reto além do limite determinado. As coordenadas Z e M não são consideradas para a simplificação.

ST_M

Se um ponto especificado não estiver associado a uma medida, ST_M poderá fornecer uma medida a ser armazenada com o ponto. Se o ponto tiver uma medida associada, ST_M poderá substituir essa medida por outra.

ST_PerpPoints

ST_PerpPoints utiliza uma curva ou multicurva e um ponto como parâmetros de entrada e retorna a projeção perpendicular do ponto especificado na curva ou multicurva. O ponto com a menor distância entre o ponto especificado e o ponto perpendicular é retornado. Se dois ou mais desses pontos projetados perpendiculares forem eqüidistantes do ponto especificado, serão todos retornados.

ST_RemovePoint

ST_RemovePoint utiliza uma curva e um ponto como parâmetros de entrada e retorna a curva especificada com todos os pontos iguais ao ponto especificado removido dela. Se a curva especificada tiver coordenadas Z ou M, o ponto também deverá ter coordenadas Z ou M.

ST_X

ST_X pode substituir a coordenada X de um ponto por outra coordenada X.

ST_Y

ST_Y pode substituir a coordenada Y de um ponto por outra coordenada Y.

ST_Z

Se um ponto especificado não tiver coordenada Z, ST_Z poderá incluir uma coordenada Z no ponto. Se o ponto tiver uma coordenada Z, ST_Z poderá substituir essa coordenada por outra coordenada Z.

Função que Retorna Informações sobre Distância

ST_Distance utiliza duas geometrias e, opcionalmente, uma unidade como parâmetros de entrada e retorna a distância mais curta entre qualquer ponto da primeira geometria até qualquer ponto da segunda geometria, medido nas unidades fornecidas.

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Se uma das duas geometrias for um valor nulo ou for vazia, será retornado um valor nulo.

Por exemplo, ST_Distance poderia informar a distância mais curta que uma aeronave deve viajar entre duas localizações. A Figura 57 ilustra essas informações.

A figura mostra um mapa dos Estados Unidos com uma linha reta entre os pontos rotulados Los Angeles e Chicago.

Figura 57. Distância Mínima Entre Duas Cidades. ST_Distance pode ter as coordenadas para as localizações de Los Angeles e Chicago como entrada e retornar um valor indicando a distância mínima entre essas localizações.

Função que Retorna Informações sobre Índice

ST_GetIndexParms utiliza o identificador para um índice espacial ou para uma coluna espacial como parâmetro de entrada e retorna os parâmetros utilizados para definir o índice ou o índice na coluna espacial. Se um número de parâmetros adicionais for especificado, apenas o parâmetro identificado pelo número será retornado.

Conversões entre Sistemas Coordenados

ST_Transform utiliza uma geometria e um identificador de sistema de referência espacial como parâmetros de entrada e transforma a geometria a ser representada no sistema de referência espacial fornecido. As projeções e conversões entre sistemas de coordenadas diferentes são executadas e as coordenadas das geometrias são acertadas adequadamente.

Capítulo 23. Funções Espaciais: Sintaxe e Parâmetros

Esta seção apresenta as funções espaciais descritas nas seções seguintes. Descreve determinados fatores que são comuns a todas, ou à maioria das funções espaciais. As funções são documentadas em ordem alfabética.

Funções Espaciais: Considerações e Tipos de Dados Associados

Esta seção fornece informações necessárias para a codificação de funções espaciais. Essas informações incluem:

- Fatores a serem considerados: os requisitos para especificar o esquema ao qual as funções espaciais pertencem e o fato de que algumas funções podem ser chamadas como métodos.
- Como tratar uma situação na qual uma função espacial não pode processar o tipo de geometria retornada por outra função espacial.
- Uma tabela mostrando quais funções assumem valores de cada tipo de dados espacial como entrada.

Fatores a Serem Considerados

Ao utilizar funções espaciais, esteja ciente destes fatores:

- Antes de chamar uma função espacial, seu nome deve ser qualificado pelo nome do esquema ao qual as funções espaciais pertencem: DB2GSE. Uma forma de fazer isto é especificar explicitamente o esquema na instrução SQL que faz referência à função, por exemplo:

  ```
  SELECT db2gse.ST_Relate (g1, g2, 'T*F**FFF2') EQUALS FROM relate_test
  ```

  Como opção, para evitar especificar o esquema sempre que uma função deve ser chamada, você pode incluir DB2GSE no registro especial CURRENT FUNCTION PATH. Para obter as definições atuais desse registro especial, digite o seguinte comando SQL:

  ```
  VALUES CURRENT FUNCTION PATH
  ```

  Para atualizar o registro especial CURRENT FUNCTION com DB2GSE, emita o seguinte comando SQL:

  ```
  set CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
  ```

- Algumas funções espaciais podem ser chamadas como métodos. No código a seguir, por exemplo, ST_Area é chamado primeiro como função e depois como um método. Nos dois casos, ST_Area é codificado para operar em um polígono que tem ID 10 e que é armazenado na coluna ZONA_DE_VENDAS de uma tabela denominada LOJAS. Quando chamado, ST_Area retorna a área do recurso real — Zona de Vendas no. 10 — que o polígono representa.

  ST_Area chamou como função:

  ```
  SELECT ST_Area(sales_zone)
  FROM   stores
  WHERE  id = 10
  ```

  ST_Area chamou como um método:

  ```
  SELECT sales_zone..ST_Area()
  FROM   stores
  WHERE  id = 10
  ```

Tratando Valores de ST_Geometry como Valores de um Subtipo

Se uma função espacial retorna uma geometria cujo tipo estático é um tipo super e se a geometria for transmitida para uma função que aceita somente geometrias de um tipo subordinado a esse tipo super, surgirá uma exceção de tempo de compilação.

Por exemplo, o tipo estático do parâmetro de saída da função ST_Union é ST_Geometry, o tipo super de todos os tipos de dados espaciais. O parâmetro de entrada estático da função ST_PointOnSurface pode ser ST_Polygon ou ST_MultiPolygon, dois subtipos de ST_Geometry. Se DB2® tentar transmitir geometrias retornadas por ST_Union para ST_PointOnSurface, o DB2 emite a seguinte exceção de tempo de compilação:

```
SQL00440N Não foi localizada função com o nome "ST_POINTONSURFACE"
com argumentos compatíveis no caminho da
função.  SQLSTATE=42884
```

Esta mensagem indica que o DB2 não pôde localizar uma função denominada ST_PointOnSurface e que tem um parâmetro de entrada ST_Geometry.

Para permitir que as geometrias do tipo super transmitam funções que aceitem somente subtipos do tipo super, utilize o operador TREAT. Conforme indicado anteriormente, ST_Union retorna geometrias de um tipo estático ST_Geometry. Também podem ser retornadas geometrias de um subtipo dinâmico de ST_Geometry. Suponha, por exemplo, que seja retornada uma geometria com um tipo dinâmico ST_MultiPolygon. Nesse caso, o operador TREAT requer que essa geometria seja utilizada com o tipo estático ST_MultiPolygon. Isto corresponde a um dos tipos de dados do parâmetro de entrada ST_PointOnSurface. Se ST_Union não retornar um valor ST_MultiPolygon, o DB2 emite uma exceção de tempo de execução.

Se uma função retornar uma geometria do tipo super, o operador TREAT poderá solicitar que o DB2 considere essa geometria como um subtipo desse tipo super. Mas esteja ciente de que essa operação é bem-sucedida somente se o subtipo corresponder ou for subordinado a um subtipo estático definido como um parâmetro de entrada da função para a qual a geometria é transmitida. Se essa condição não for atendida, o DB2 emite uma exceção de tempo de execução.

Considere outro exemplo: suponha que você deseje determinar os pontos perpendiculares para um determinado ponto no limite de um polígono que não tem orifícios. Utilize a função ST_Boundary para derivar o limite do polígono. O parâmetro de saída estático de ST_Boundary é ST_Geometry, mas ST_PerpPoints aceita geometrias ST_Curve. Como todos os polígonos têm uma cadeia de linha (que também é uma curva) como limite e como o tipo de dados das cadeias de linhas (ST_LineString) é subordinado a ST_Curve, a operação a seguir permite que um polígono ST_Geometry retornado por ST_Boundary seja transmitido para ST_PerpPoints:

```
SELECT ST_AsText(ST_PerpPoints(TREAT(ST_Boundary(polygon) as ST_Curve)),
       ST_Point(30.5, 65.3, 1)))
FROM   polygon_table
```

Em vez de chamar ST_Boundary e ST_PerpPoints como funções, você pode chamá-los como métodos. Para isto, especifique o seguinte código:

```
SELECT TREAT(ST_Boundary(polygon) as ST_Curve)..
       ST_PerpPoints(St_Point(30.5, 65.3, ))..ST_AsText()
FROM   polygon_table
```

Funções Espaciais Relacionadas de Acordo com o Tipo de Entrada

A Tabela 56 relaciona as funções espaciais de acordo com o tipo de entrada que podem aceitar.

Importante: Como indicado em outro local, os tipos de dados espaciais formam uma hierarquia, com ST_Geometry como raiz. Quando a documentação do DB2 Spatial Extender indica que um valor do tipo super nessa hierarquia pode ser utilizado como entrada para uma função, como opção, um valor de qualquer subtipo desse tipo super também pode ser utilizado como entrada para a função.

Por exemplo, as primeiras entradas na Tabela 56 indicam que ST_Area e diversas outras funções podem assumir valores do tipo de dados ST_Geometry como entrada. Portanto, a entrada dessas funções também pode ser valores de qualquer subtipo de ST_Geometry: ST_Point, ST_Curve, ST_LineString, etc.

Tabela 56. Funções Espaciais Relacionadas de Acordo com o Tipo de Entrada

Tipo de Dados do Parâmetro de Entrada	Função
ST_Geometry	EnvelopesIntersect ST_Area ST_AsBinary ST_AsGML ST_AsShape ST_AsText ST_Boundary ST_Buffer ST_BuildMBRAggr ST_BuildUnionAggr ST_Centroid ST_Contains ST_ConvexHull ST_CoordDim ST_Crosses ST_Difference ST_Dimension ST_Disjoint ST_Distance ST_Envelope ST_EnvIntersects ST_Equals ST_FindMeasure ou ST_LocateAlong ST_Generalize ST_GeometryType

Tabela 56. Funções Espaciais Relacionadas de Acordo com o Tipo de Entrada (continuação)

Tipo de Dados do Parâmetro de Entrada	Função
ST_Geometry, continuação	ST_Intersection ST_Intersects ST_Is3D ST_IsEmpty ST_IsMeasured ST_IsSimple ST_IsValid ST_MaxM ST_MaxX ST_MaxY ST_MaxZ ST_MBR ST_MBRIntersects ST_MeasureBetween ou ST_LocateBetween ST_MinM ST_MinX ST_MinY ST_MinZ ST_NumPoints ST_Overlaps ST_Relate ST_SRID ou ST_SrsId ST_SrsName ST_SymDifference ST_ToGeomColl ST_ToLineString ST_ToMultiLine ST_ToMultiPoint ST_ToMultiPolygon ST_ToPoint ST_ToPolygon ST_Touches ST_Transform ST_Union ST_Within
ST_Point	ST_M ST_X ST_Y ST_Z
ST_Curve	ST_AppendPoint ST_ChangePoint ST_EndPoint ST_IsClosed ST_IsRing ST_Length ST_MidPoint ST_PerpPoints ST_RemovePoint ST_StartPoint
ST_LineString	ST_PointN ST_Polygon
ST_Surface	ST_Perimeter ST_PointOnSurface
ST_GeomCollection	ST_GeometryN ST_NumGeometries

Tabela 56. Funções Espaciais Relacionadas de Acordo com o Tipo de Entrada (continuação)

Tipo de Dados do Parâmetro de Entrada	Função
ST_MultiPoint	ST_PointN
ST_MultiCurve	ST_IsClosed ST_Length ST_PerpPoints
ST_MultiLineString	ST_LineStringN ST_NumLineStrings ST_Polygon
ST_MultiSurface	ST_Perimeter ST_PointOnSurface
ST_MultiPolygon	ST_NumPolygons ST_PolygonN

As funções ST_BuildMBRAggr e ST_BuildUnionAggr são descritas em ″MBR Aggregate″ e ″Union Aggregate″, respectivamente.

Referência Relacionada:

- “Agregado MBR” na página 345
- “ST_Boundary” na página 356
- “ST_Area” na página 348
- “ST_PerpPoints” na página 466
- “ST_Point” na página 468
- “ST_PointOnSurface” na página 475
- “ST_Relate” na página 482
- “ST_Union” na página 500
- “Agregado de União” na página 510

EnvelopesIntersect

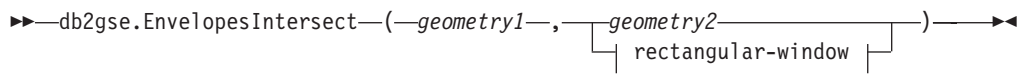
EnvelopesIntersect aceita dois tipos de parâmetros de entrada:

- Duas geometrias

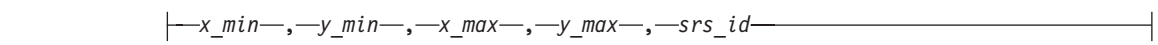
  EnvelopesIntersect retornará 1 se o envelope da primeira geometria cruzar com o envelope da segunda.Caso contrário, será retornado 0 (zero).
- Uma geometria, quatro valores de coordenadas do tipo DOUBLE que definem os cantos inferior esquerdo e superior direito de uma janela retangular e o identificador do sistema de referência espacial.

  EnvelopesIntersect retorna 1 se o envelope da primeira geometria fizer interseção com o envelope definido pelos quatro valores do tipo DOUBLE. Caso contrário, será retornado 0 (zero).

Sintaxe:



rectangular-window:



Parâmetros:

geometry1
Um valor do tipo ST_Geometry ou um dos seus subtipos que representam a geometria cujo envelope será testado quanto à interseção com o envelope de *geometry2* ou com a janela retangular definida pelos quatro valores do tipo DOUBLE.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria cujo envelope será testado quanto à interseção com o envelope de *geometry1*.

x_min
Especifica o valor mínimo da coordenada X para o envelope. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

Para dados geodésicos, as seguintes condições se aplicam:

- *x_min* deve ser um valor de longitude entre –180 e 180 graus.
- *x_min* é maior que *x_max* quando o envelope sobrepõe o meridiano de 180 graus.

y_min
Especifica o valor mínimo da coordenada Y para o envelope. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

Para dados geodésicos, as seguintes condições se aplicam:

- *y_min* deve ser um valor de latitude entre –90 e 90 graus.
- *y_min* deve ser menor que o valor *y_max*.

x_max
Especifica o valor máximo da coordenada X para o envelope. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

Para dados geodésicos, as seguintes condições se aplicam:

- *x_max* deve ser um valor de longitude entre –180 e 180 graus.
- *x_max* é menor que o valor *x_min* quando o envelope sobrepõe o meridiano de 180 graus.

y_max
Especifica o valor máximo da coordenada Y para o envelope. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é DOUBLE.

Para dados geodésicos, as seguintes condições se aplicam:

- *y_max* deve ser um valor de latitude entre –90 e 90 graus.
- *y_max* deve ser maior que o valor *y_min*.

srs_id
Identifica exclusivamente o sistema de referência espacial. O identificador do

sistema de referência espacial deve corresponder ao identificador do sistema de referência espacial do parâmetro de geometria. Você deve especificar um valor diferente de nulo para este parâmetro.

O tipo de dados deste parâmetro é INTEGER.

Tipo de Retorno:

INTEGER

Exemplo:

Este exemplo cria dois polígonos que representam regiões e depois determina se uma delas faz interseção com uma área geográfica especificada pelos quatro valores do tipo DOUBLE.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE counties (id INTEGER, name CHAR(20), geometry ST_Polygon)

INSERT INTO counties VALUES
       (1, 'County_1', ST_Polygon('polygon((0 0, 30 0, 40 30, 40 35,
        5 35, 5 10, 20 10, 20 5, 0 0))' ,0))

INSERT INTO counties VALUES
       (2, 'County_2', ST_Polygon('polygon((15 15, 15 20, 60 20, 60 15,
        15 15))' ,0))

INSERT INTO counties VALUES
       (3, 'County_3', ST_Polygon('polygon((115 15, 115 20, 160 20, 160 15,
        115 15))' ,0))


SELECT name
FROM counties as c
WHERE EnvelopesIntersect(c.geometry, 15, 15, 60, 20, 0) =1
```

Resultados:

```
Nome--------------------
County_1
County_2
```

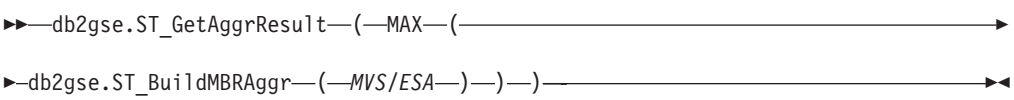
Agregado MBR

A combinação das funções ST_BuildMBRAggr e ST_GetAggrResult agrega uma coluna de geometrias em uma coluna selecionada em uma única geometria, construindo um retângulo que representa o retângulo de limite mínimo que inclui todas as geometrias na coluna. As coordenadas Z e M são descartadas quando o agregado é calculado.

Se todas as geometrias a serem combinadas forem nulas, será retornado nulo. Se todas as geometrias forem nulas ou vazias, será retornada uma geometria vazia. Se o retângulo limite mínimo de todas as geometrias a serem combinadas resultar em um ponto, este ponto será retornado como valor ST_Point. Se o retângulo limite mínimo de todas as geometrias a serem combinadas resultar em uma cadeia de linha horizontal ou vertical, essa cadeia de linha será retornada como um valor ST_LineString. Caso contrário, o retângulo limite mínimo será retornado como um valor ST_Polygon.

Agregado MBR

Sintaxe:



Parâmetro:

geometrias
Uma coluna selecionada do tipo ST_Geometry ou um dos seus subtipos e que representa todas as geometrias para as quais o retângulo limite mínimo deve ser calculado.

Tipo de Retorno:

db2gse.ST_Geometry

Restrições:

Não é possível construir o agregado de união de uma coluna espacial em uma seleção completa em qualquer uma das seguintes situações:

- Em um ambiente MPP
- Se a cláusula GROUP BY for utilizada na seleção completa
- Se você utilizar uma função diferente da função agregada do DB2 MAX.

Exemplo:

No exemplo a seguir, as linhas de resultados foram reformatadas para facilitar a leitura. O espaçamento nos resultados varia de acordo com a exibição on-line.

Este exemplo mostra como utilizar a função ST_BuildMBRAggr para obter o retângulo limite máximo de todas as geometrias dentro de uma coluna. Neste exemplo, diversos pontos são incluídos na coluna GEOMETRY na tabela SAMPLE_POINTS. Em seguida, o código SQL determina o retângulo limite máximo de todos os pontos juntos.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_points (id integer, geometry ST_Point)

INSERT INTO sample_points (id, geometry)
VALUES
    (1, ST_Point(2,   3, 1)),
    (2, ST_Point(4,   5, 1)),
    (3, ST_Point(13, 15, 1)),
    (4, ST_Point(12,  5, 1)),
    (5, ST_Point(23,  2, 1)),
    (6, ST_Point(11,  4, 1))


SELECT cast(ST_GetAggrResult(MAX(ST_BuildMBRAggr
    (geometry)))..ST_AsText AS varchar(160))
    AS ";Aggregate_of_Points";
FROM sample_points
```

Resultados:

```
Aggregate_of_Points
------------------------------------------------------
POLYGON (( 2.00000000 2.00000000, 23.00000000 2.00000000,
23.00000000 15.00000000, 2.00000000 15.00000000, 2.00000000
2.00000000))
```

ST_AppendPoint

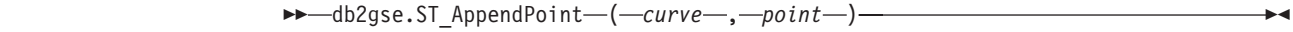
ST_AppendPoint utiliza uma curva e um ponto como parâmetros de entrada e estende a curva pelo ponto especificado. Se a curva especificada tiver coordenadas Z ou M, o ponto também deverá ter coordenadas Z ou M. A curva resultante é representada no sistema de referência espacial da curva especificada.

Se o ponto a ser anexado não for representado no mesmo sistema de referência espacial que a curva, ele será convertido para o outro sistema de referência espacial.

Se a curva especificada for fechada ou simples, a curva resultante pode não ser mais fechada ou simples. Se a curva ou ponto especificado for nulo, ou se a curva for vazia, será retornado nulo. Se o ponto a ser anexado for vazio, a curva especificada será retornada inalterada e um aviso será retornado (SQLSTATE 01HS3).

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

curve Um valor do tipo ST_Curve ou um de seus subtipos que representa a curva à qual *point* será anexado.

point Um valor do tipo ST_Point que representa o ponto que está anexado a *curve*.

Tipo de Retorno:

db2gse.ST_Curve

Exemplos:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este código cria duas cadeias de linhas, cada uma com três pontos.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines(id integer, line ST_Linestring)

INSERT INTO sample_lines VALUES
    (1, ST_LineString('linestring (10 10, 10 0, 0 0 )', 0) )

INSERT INTO sample_lines VALUES
    (2, ST_LineString('linestring z (0 0 4, 5 5 5, 10 10 6)', 0) )
```

Exemplo 1:

Este exemplo inclui o ponto (5, 5) no final de uma cadeia de linhas.

```
SELECT CAST(ST_AsText(ST_AppendPoint(line, ST_Point(5, 5)))
    AS VARCHAR(120)) New
  FROM sample_linesWHERE  id=1
```

Resultados:

```
NEW-------------------------------------------------------------
LINESTRING ( 10.00000000 10.00000000, 10.00000000 0.00000000,
0.00000000 0.00000000, 5.00000000 5.00000000)
```

Exemplo 2:

Este exemplo inclui o ponto (15, 15, 7) no final de uma cadeia de linhas com coordenadas Z.

```
SELECT CAST(ST_AsText(ST_AppendPoint(line, ST_Point(15.0, 15.0, 7.0)))
   AS VARCHAR(160)) New
  FROM sample_linesWHERE  id=2
```

Resultados:

```
NEW-------------------------------------------------------------------
LINESTRING Z ( 0.00000000 0.00000000 4.00000000, 5.00000000
5.00000000 5.00000000, 10.00000000 10.00000000 6.00000000,
15.00000000 15.00000000 7.00000000)
```

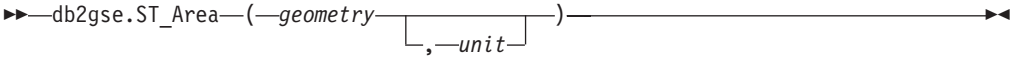
ST_Area

ST_Area utiliza uma geometria e, opcionalmente, uma unidade como parâmetros de entrada e retorna a área coberta pela geometria na unidade de medida padrão ou especificada.

Se a geometria for um polígono ou multipolígono, a área coberta pela geometria será retornada. A área de pontos, cadeias de linhas, multipontos e cadeias multilinha é 0 (zero). Se a geometria for nula ou vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que determina a área.

unit Um valor VARCHAR(128) que identifica as unidades nas quais a área é medida. As unidades de medida suportadas estão listadas na exibição do catálogo DB2GSE.ST_UNITS_OF_MEASURE.

Se o parâmetro *unit* for omitido, as seguintes regras serão utilizadas para determinar a unidade na qual a área será medida:

- Se *geometry* estiver em um sistema de coordenadas projetadas ou geocêntricas, a unidade linear associada a este sistema de coordenadas será utilizada.

- Se *geometry* estiver em um sistema de coordenadas geográficas, mas não estiver em um SRS (spatial reference system, sistema de referência espacial) geodésico, a unidade angular associada a este sistema de coordenadas será utilizada.
- Se *geometry* estiver em um SRS geodésico, a unidade de medida padrão será metros quadrados.

Restrições para conversões de unidades: Será retornado um erro (SQLSTATE 38SU4) se ocorrer qualquer uma das condições a seguir:

- A geometria está em um sistema de coordenadas não especificado e o parâmetro *unit* é especificado.
- A geometria está em um sistema de coordenadas projetadas e uma unidade angular é especificada.
- A geometria está em um sistema de coordenadas geográficas, mas não está em um SRS geodésico e uma unidade linear é especificada.
- A geometria está em um sistema de coordenadas geográficas, está em um SRS geodésico e uma unidade angular é especificada.

Tipo de Retorno:

DOUBLE

Exemplos:

Exemplo 1:

O analista espacial precisa de uma lista da área coberta por cada região de vendas. Os polígonos da região de vendas são armazenados na tabela SAMPLE_POLYGONS. A área é calculada pela aplicação da função ST_Area na coluna da geometria.

```
db2se create_srs se_bank  -srsId 4000 -srsName new_york1983 -xOffset 0
    -yOffset 0 -xScale 1 -yScale 1
    -coordsysName NAD_1983_StatePlane_New_York_East_FIPS_3101_Feet

CREATE TABLE sample_polygons (id INTEGER, geometry ST_POLYGON)

INSERT INTO sample_polygons (id, geometry)
VALUES
  (1, ST_Polygon('polygon((0 0, 0 10, 10 10, 10 0, 0 0))', 4000) ),
  (2, ST_Polygon('polygon((20 0, 30 20, 40 0, 20 0 ))', 4000) ),
  (3, ST_Polygon('polygon((20 30, 25 35, 30 30, 20 30))', 4000))
```

A instrução SELECT a seguir recupera o ID e a área da região de vendas:

```
SELECT id, ST_Area(geometry) AS area
FROM   sample_polygons
```

Resultados:

```
ID        AREA
--------- ------------------------
       1   +1.00000000000000E+002
       2   +2.00000000000000E+002
       3   +2.50000000000000E+001
```

Exemplo 2:

A instrução SELECT a seguir recupera o ID e a área da região de vendas em várias unidades:

```
SELECT id,
       ST_Area(geometry) square_feet,
       ST_Area(geometry, 'METER') square_meters,
       ST_Area(geometry, 'STATUTE MILE') square_miles
FROM   sample_polygons
```

Resultados:

```
ID  SQUARE_FEET              SQUARE_METERS            SQUARE_MILES
--- ------------------------ ------------------------ ------------------------
  1 +1.00000000000000E+002   +9.29034116132748E+000   +3.58702077598427E-006
  2 +2.00000000000000E+002   +1.85806823226550E+001   +7.17404155196855E-006
  3 +2.50000000000000E+001   +2.32258529033187E+000   +8.96755193996069E-007
```

Exemplo 3:

Este exemplo encontra a área de um polígono definida nas coordenadas de Plano de Estado.

O sistema de referência espacial Plano de Estado com um ID 3 é criado com o seguinte comando:

```
db2se create_srs SAMP_DB  -srsId 3 -srsName z3101a -xOffset 0
    -yOffset 0 -xScale 1 -yScale 1
    -coordsysName NAD_1983_StatePlane_New_York_East_FIPS_3101_Feet
```

As seguintes instruções SQL adicionam o polígono, no sistema de referência espacial 3, à tabela e determinam a área em pés quadrados, em metros quadrados e em milhas quadradas.

```
SET current function path db2gse;
 CREATE TABLE Sample_Poly3 (id integer, geometry ST_Polygon);
 INSERT into Sample_Poly3 VALUES
    (1, ST_Polygon('polygon((567176.0 1166411.0,
                             567176.0 1177640.0,
                             637948.0 1177640.0,
                             637948.0 1166411.0,
                             567176.0 1166411.0 ))', 3));
 SELECT id, ST_Area(geometry) "Square Feet",
            ST_Area(geometry, 'METER') "Square Meters",
            ST_Area(geometry, 'STATUTE MILE') "Square Miles"
 FROM Sample_Poly3;
```

Resultados:

```
ID   Square Feet              Square Meters            Square Miles
--- ------------------------ ------------------------ ------------------------
  1 +7.94698788000000E+008   +7.38302286101346E+007   +2.85060106320552E+001
```

Exemplo 4:

O analista espacial precisa de uma lista da área coberta por cada região de exploração. Os polígonos da região de exploração estão armazenados na tabela SAMPLE_GEODETIC_TAB e incluem as seguintes regiões:

- Uma região ao redor do Pólo Norte
- Uma região ao redor do Pólo Sul
- Uma região que estende o Meridiano de 180 graus

O segundo campo no seguinte arquivo de entrada, samp_wkt_rows.txt, contém polígonos que representam estas regiões:

```
1|'polygon((5 82,15 82,25 82,35 82,45 82,55 82,65 82,75 82,85 82,95 82,
105 82,115 82,125 82,135 82,145 82,155 82,165 82,175 82,-175 82,-165 82,
-155 82,-145 82,-135 82,-125 82,-115 82,-105 82,-95 82,-85 82,-75 82,
-65 82,-55 82,-45 82,-35 82,-25 82,-15 82,-5 82,5 82))'|'North Pole region'
2|'polygon((175 -82,165 -82,155 -82,145 -82,135 -82,125 -82,115 -82,
105 -82,95 -82,85 -82,75 -82,65 -82,55 -82,45 -82,35 -82,25 -82,15 -82,
5 -82,-5 -82,-15 -82,-25 -82,-35 -82,-45 -82,-55 -82,-65 -82,-75 -82,
-85 -82,-95 -82,-105 -82,-115 -82,-125 -82,-135 -82,-145 -82,-155 -82,
-165 -82,-175 -82,175 -82))'|'South Pole region'
3|'polygon((-175 -42,-175 1,-175 42,175 42,175 -1,175 -42,-175 -42))
'|'180th meridian'
```

As seguintes instruções SQL adicionam os polígonos, em um sistema de referência espacial geodésico 2000000000, à tabela SAMPLE_GEODETIC_TAB.

```
SET current function path db2gse;
CREATE TABLE db2se_samp.gsege_temp_samp (
            gid      INTEGER,
            g1_wkt   varchar(500),
            comment  varchar(255)
   ) NOT LOGGED INITIALLY;
LOAD FROM samp_wkt_rows.txt OF DEL MODIFIED BY CHARDEL'' COLDEL|
   INSERT INTO db2se_samp.gsege_temp_samp;

CREATE TABLE sample_geodetic_tab
  (gid INTEGER NOT NULL PRIMARY KEY,
   geometry ST_Geometry),
   comment varchar(255));

INSERT INTO sample_geodetic_tab
     SELECT gid, ST_GeomFromText(g1_wkt, 2000000000), comment
   FROM db2se_samp.gsege_temp_samp;
```

A função ST_Area calcula a área do polígono na coluna da geometria. A unidade de medida padrão de ST_Area é metros quadrados. A instrução SELECT a seguir recupera o ID e a área da região de exploração em metros quadrados, em pés quadrados e em linhas quadradas.

```
SELECT id, ST_Area(geometry) AS SQUARE_METERS,
ST_Area(geometry,'FOOT') AS SQUARE_FEET,
ST_Area(geometry, 'STATUTE MILE') AS SQUARE_MILES
FROM sample_geodetic_tab
WHERE id BETWEEN 1 AND 9 ORDER BY id;
```

```
ID  SQUARE_METERS           SQUARE_FEET             SQUARE_MILES
--- ----------------------  ----------------------  ----------------------
  1 +2.52472719957839E+012  +2.71759374028922E+013  +9.74802621488040E+005
  2 +2.52475431563494E+012  +2.71762292776957E+013  +9.74813091056005E+005
  3 +9.43568029137069E+012  +1.01564817377028E+014  +3.64313652781464E+006
```

Referência Relacionada:

- "Funções Espaciais Suportadas pelo DB2 Geodetic Extender" na página 208
- "A Exibição do Catálogo DB2GSE.ST_UNITS_OF_MEASURE" na página 298

ST_AsBinary

ST_AsBinary utiliza uma geometria como um parâmetro de entrada e retorna sua representação binária reconhecida. As coordenadas Z e M são descartadas e não farão parte da representação binária reconhecida.

Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos a serem convertidos para a representação binária reconhecida correspondente.

Tipo de Retorno:

BLOB(2G)

Exemplos:

O código a seguir ilustra como utilizar a função ST_AsBinary para converter os pontos nas colunas da geometria da tabela SAMPLE_POINTS em representação binária reconhecida (WKB) na coluna BLOB.

```
CREATE TABLE SAMPLE_POINTS (id integer, geometry ST_POINT, wkb BLOB(32K))

INSERT INTO SAMPLE_POINTS (id, geometry)
VALUES
    (1100, ST_Point(10, 20, 1))
```

Exemplo 1:

Este exemplo ocupa a coluna WKB com um ID 1111, a partir da coluna GEOMETRY, com um ID 1100.

```
INSERT INTO sample_points(id, wkb)
VALUES (1111,
  (SELECT  ST_AsBinary(geometry)
   FROM    sample_points
   WHERE   id = 1100))

SELECT id, cast(ST_Point(wkb)..ST_AsText AS varchar(35)) AS point
FROM   sample_points
WHERE  id = 1111
```

Resultados:

```
ID          Point
----------- -----------------------------------
       1111 POINT ( 10.00000000 20.00000000)
```

Exemplo 2:

Este exemplo exibe a representação binária WKB.

```
SELECT id, substr(ST_AsBinary(geometry), 1, 21) AS point_wkb
FROM   sample_points
WHERE  id = 1100
```

Resultados:

```
ID     POINT_WKB
------ ---------------------------------------------
  1100 x'0101000000000000000000244000000000000034400'
```

Referência Relacionada:

- "Representação WKB (Well-Known Binary)" na página 524

ST_AsGML

ST_AsGML utiliza uma geometria como parâmetro de entrada e retorna sua representação utilizando a linguagem de marcação geográfica.

Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos a serem convertidos na representação GML correspondente.

Tipo de Retorno:

CLOB(2G)

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O fragmento de código a seguir ilustra como utilizar a função ST_AsGML para exibir o fragmento de GML. Este exemplo ocupa a coluna GML, a partir da coluna da geometria, com um ID 2222.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE SAMPLE_POINTS (id integer, geometry ST_POINT, gml CLOB(32K))

INSERT INTO SAMPLE_POINTS (id, geometry)
VALUES
    (1100, ST_Point(10, 20, 1))

INSERT INTO sample_points(id, gml)
VALUES (2222,
  (SELECT  ST_AsGML(geometry)
   FROM    sample_points
   WHERE   id = 1100))
```

A instrução SELECT a seguir lista o ID e a representação GML das geometrias. A geometria é convertida em um fragmento de GML pela função ST_AsGML.

```
SELECT id, cast(ST_AsGML(geometry) AS varchar(110)) AS gml_fragment
FROM   sample_points
WHERE  id = 1100
```

Resultados:

```
A instrução SELECT retorna o seguinte conjunto de resultados:

ID          GML_FRAGMENT

----------- ------------------------------------------------------------
```

```
1100 <gml:Point srsName";EPSG:4269";><gml:coord>
     <gml:X>10</gml:X><gml:Y>20</gml:Y>
     </gml:coord></gml:Point>
```

Referência Relacionada:

- "Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados" na página 299
- "Representação GML (Geography Markup Language)" na página 526

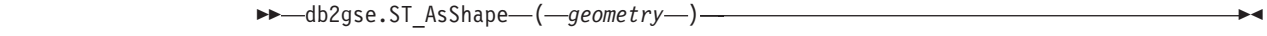
ST_AsShape

ST_AsShape utiliza uma geometria como um parâmetro de entrada e retorna sua representação de formato ESRI.

Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos a serem convertidos para a representação de formato ESRI correspondente.

Return type:

BLOB(2G)

Exemplo:

O fragmento de código a seguir ilustra como utilizar a função ST_AsShape para converter os pontos na coluna da geometria da tabela SAMPLE_POINTS em representação binária de formatos na coluna BLOB de formato. Este exemplo ocupa a coluna de formato a partir da coluna da geometria. A representação binária de formatos é utilizada para exibir as geometrias em geonavegadores, que requerem que as geometrias estejam em conformidade com o formato do arquivo modelo ESRI, ou que as geometrias sejam construídas para o arquivo *.SHP do arquivo modelo.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE SAMPLE_POINTS (id integer, geometry ST_POINT, shape BLOB(32K))

INSERT INTO SAMPLE_POINTS (id, geometry)
VALUES
    (1100, ST_Point(10, 20, 1))

INSERT INTO sample_points(id, shape)
VALUES (2222,
  (SELECT  ST_AsShape(geometry)
   FROM    sample_points
   WHERE   id = 1100))
```

```
SELECT id, substr(ST_AsShape(geometry), 1, 20) AS shape
FROM   sample_points
WHERE  id = 1100
```

Retorna:

```
ID     SHAPE

------ ----------------------------------------
  1100 x'0100000000000000000002440000000000003440'
```

Referência Relacionada:

- "Representação de Formatos" na página 526

ST_AsText

ST_AsText utiliza uma geometria como um parâmetro de entrada e retorna sua representação de texto reconhecida.

Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos a serem convertidos para a representação de texto reconhecida correspondente.

Return type:

CLOB(2G)

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Depois de capturar e inserir dados na tabela SAMPLE_GEOMETRIES, um analista deseja verificar se os valores inseridos estão corretos, examinando a representação de texto reconhecida das geometrias.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries(id SMALLINT, spatial_type varchar(18),
    geometry ST_GEOMETRY)

INSERT INTO sample_geometries(id, spatial_type, geometry)
VALUES
    (1, 'st_point', ST_Point(50, 50, 0)),
    (2, 'st_linestring', ST_LineString('linestring
        (200 100, 210 130, 220 140)',  0)),
    (3, 'st_polygon', ST_Polygon('polygon((110 120, 110 140,
        130 140, 130 120, 110 120))', 0))
```

A instrução SELECT a seguir lista o tipo espacial e a representação WKT das geometrias. A geometria é convertida em texto pela função ST_AsText. Então é feita uma conversão em um varchar(120) porque a saída padrão da função ST_AsText é CLOB(2G).

```
SELECT id, spatial_type, cast(geometry..ST_AsText
   AS varchar(150)) AS wkt
FROM   sample_geometries
```

Resultados:

```
ID     SPATIAL_TYPE       WKT
------ ------------------ -----------------------------------------------
     1 st_point           POINT ( 50.00000000 50.00000000)

     2 st_linestring      LINESTRING ( 200.00000000 100.00000000,
                            210.00000000 130.00000000, 220.00000000
                            140.00000000)

     3 st_polygon         POLYGON (( 110.00000000 120.00000000,
                            130.00000000 120.00000000, 130.00000000
                            140.00000000, 110.00000000140.00000000,
                            110.00000000 120.00000000))
```

Referência Relacionada:

- "Representação WKT (Well-Known Text)" na página 519

ST_Boundary

ST_Boundary utiliza uma geometria como um parâmetro de entrada e retorna seu limite como uma nova geometria. A geometria resultante é representada no sistema de referência espacial da geometria especificada.

Se a geometria especificada for um ponto, multiponto, curva fechada ou multicurva fechada, ou se for fazia, o resultado será uma geometria vazia do tipo ST_Point. Para curvas ou multicurvas que não são fechadas, os pontos iniciais e finais das curvas são retornados como um valor ST_MultiPoint, a menos que esse ponto seja o ponto inicial ou o final de um número de curvas par. Para superfícies e multisuperfícies, a curva que define o limite da geometria especificada é retornada como um valor ST_Curve ou ST_MultiCurve. Se a geometria especificada for nula, então nulo é retornado.

Se possível, o tipo específico da geometria retornada será ST_Point, ST_LineString ou ST_Polygon. Por exemplo, o limite de um polígono sem aberturas é uma cadeia de linhas única, representada como ST_LineString. O limite de um polígono com uma ou mais aberturas consiste em várias cadeias de linhas, representadas como ST_MultiLineString.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos. O limite desta geometria é retornado.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo cria várias geometrias e determina o limite de cada uma delas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Polygon('polygon((40 120, 90 120, 90 150, 40 150, 40 120))', 0))

INSERT INTO sample_geoms VALUES
       (2, ST_Polygon('polygon((40 120, 90 120, 90 150, 40 150, 40 120),
                               (70 130, 80 130, 80 140, 70 140, 70 130))' ,0))

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('linestring(60 60, 65 60, 65 70, 70 70)' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('multilinestring((60 60, 65 60, 65 70, 70 70),
                                        (80 80, 85 80, 85 90, 90 90),
                                        (50 50, 55 50, 55 60, 60 60))' ,0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('point(30 30)' ,0))

SELECT id, CAST(ST_AsText(ST_Boundary(geometry)) as VARCHAR(320)) Boundary
FROM   sample_geoms

Resultados
ID      BOUNDARY
------- ------------------------------------------------------------------------
      1 LINESTRING ( 40.00000000 120.00000000, 90.00000000 120.00000000,
          90.00000000 150.00000000, 40.00000000 150.00000000, 40.00000000
          120.00000000)

      2 MULTILINESTRING (( 40.00000000 120.00000000, 90.00000000 120.00000000,
          90.00000000 150.00000000, 40.00000000 150.00000000, 40.00000000
          120.00000000),( 70.00000000 130.00000000, 70.00000000 140.00000000,
          80.00000000 140.00000000, 80.00000000 130.00000000, 70.00000000
          130.00000000))

      3 MULTIPOINT ( 60.00000000 60.00000000, 70.00000000 70.00000000)

      4 MULTIPOINT ( 50.00000000 50.00000000, 70.00000000 70.00000000,
          80.00000000 80.00000000, 90.00000000 90.00000000)

      5 POINT EMPTY
```

ST_Buffer

ST_Buffer utiliza uma geometria, uma distância e, opcionalmente, uma unidade como parâmetros de entrada e retorna a geometria que engloba a geometria especificada pela distância especificada, medida na unidade especificada. Cada ponto no limite da geometria resultante representa a distância especificada da geometria especificada. A geometria resultante é representada no sistema de referência espacial da geometria especificada.

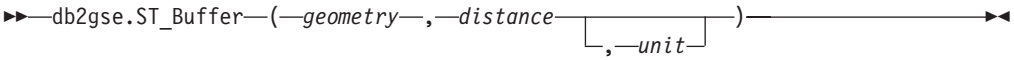
Para dados geodésicos, se você especificar uma distância negativa, ST_Buffer retornará uma região que está mais além da distância especificada de todos os pontos da geometria de entrada. Em outras palavras, a distância retorna a região complementar.

Qualquer curva circular no limite da geometria resultante é aproximada por cadeias lineares. Por exemplo, o buffer em torno de um ponto, que pode resultar em uma região circular, é aproximado por um polígono cujo limite é uma cadeia de linhas.

Se a geometria especificada for nula ou estiver vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria para criar o buffer adjacente. Para dados geodésicos, ST_Buffer suporta apenas os tipos de dados ST_Point e ST_MultiPoint.

distance
Um valor DOUBLE PRECISION que especifica a distância a ser utilizada para o buffer em torno da *geometria*.Para dados geodésicos, a distância não deve ser maior do que o raio equatorial da Terra. Para o elipsóide WGS-84, este comprimento é de 6378137.0 metros.

unit Um valor VARCHAR(128) que identifica a unidade na qual a *distância* é calculada. As unidades de medida suportadas estão listadas na exibição do catálogo DB2GSE.ST_UNITS_OF_MEASURE.

Se o parâmetro *unit* for omitido, as seguintes regras serão utilizadas para determinar a unidade de medida utilizada para a distância:

- Se *geometry* estiver em um sistema de coordenadas projetadas ou geocêntricas, a unidade linear associada a este sistema de coordenadas será a padrão.
- Se *geometry* estiver em um sistema de coordenadas geográficas, mas não estiver em um SRS (Spatial Reference System, sistema de referência espacial) geodésico, a unidade angular associada a este sistema de coordenadas será a padrão.
- Se *geometry* estiver em um SRS geodésico, a unidade de medida padrão será metros.

Restrições para conversões de unidades: Será retornado um erro (SQLSTATE 38SU4) se ocorrer qualquer uma das condições a seguir:

- A geometria está em um sistema de coordenadas não especificado e o parâmetro *unit* é especificado.
- A geometria está em um sistema de coordenadas projetadas e uma unidade angular é especificada.
- A geometria está em um sistema de coordenadas geográficas, mas não está em um SRS geodésico e uma unidade linear é especificada.
- A geometria está em um sistema de coordenadas geográficas, está em um SRS geodésico e uma unidade angular é especificada.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplos:

Nos exemplos a seguir, os resultados foram reformatados para possibilitar a leitura. O espaçamento nos resultados irá variar de acordo com a exibição.

O código a seguir cria um sistema de referência espacial, cria a tabela SAMPLE_GEOMETRIES e ocupa-a.

```
db2se create_srs se_bank -srsId 4000 -srsName new_york1983
  -xOffset 0 -yOffset 0 -xScale 1 -yScale 1
  -coordsysName NAD_1983_StatePlane_New_York_East_FIPS_3101_Feet
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE
    sample_geometries (id INTEGER, spatial_type varchar(18),
    geometry ST_GEOMETRY)

INSERT INTO sample_geometries(id, spatial_type, geometry)
VALUES
    (1, 'st_point', ST_Point(50, 50, 4000)),
    (2, 'st_linestring',
         ST_LineString('linestring(200 100, 210 130,
         220 140)',  4000)),
    (3, 'st_polygon',
         ST_Polygon('polygon((110 120, 110 140, 130 140,
         130 120, 110 120))',4000)),
    (4, 'st_multipolygon',
         ST_MultiPolygon('multipolygon(((30 30, 30 40,
         35 40, 35 30, 30 30),(35 30, 35 40, 45 40,
         45 30, 35 30)))', 4000))
```

Exemplo 1:

A instrução SELECT a seguir utiliza a função ST_Buffer para aplicar um buffer de 10.

```
SELECT id, spatial_type,
       cast(geometry..ST_Buffer(10)..ST_AsText AS varchar(470)) AS buffer_10
FROM   sample_geometries
```

Resultados:

```
ID          SPATIAL_TYPE       BUFFER_10
----------- ------------------ ----------------------------------------
1           st_point           POLYGON (( 60.00000000 50.00000000,
   59.00000000 55.00000000, 54.00000000 59.00000000, 49.00000000
   60.00000000, 44.00000000 58.00000000, 41.00000000 53.00000000,
   40.00000000 48.00000000,42.00000000 43.00000000, 47.00000000
   41.00000000, 52.00000000 40.00000000, 57.00000000 42.00000000,
   60.00000000 50.00000000))

2             st_linestring    POLYGON (( 230.00000000
   140.00000000, 229.00000000 145.00000000, 224.00000000
   149.00000000, 219.00000000 150.00000000, 213.00000000 147.00000000,
   203.00000000 137.00000000, 201.00000000 133.00000000, 191.00000000
   103.00000000, 191.00000000 99.00000000, 192.00000000  95.00000000,
   196.00000000 91.00000000, 200.00000000 91.00000000,204.00000000
   91.00000000, 209.00000000 97.00000000, 218.00000000 124.00000000,
   227.00000000 133.00000000, 230.00000000 140.00000000))

3              st_polygon         POLYGON (( 140.00000000 120.00000000,
   140.00000000 140.00000000, 139.00000000  145.00000000, 130.00000000
```

```
   150.00000000, 110.00000000 150.00000000, 105.00000000 149.00000000,
   100.00000000 140.00000000,100.00000000 120.00000000, 101.00000000
   115.00000000, 110.00000000 110.00000000,130.00000000 110.00000000,
   135.00000000 111.00000000, 140.00000000 120.00000000))

4              st_multipolygon    POLYGON (( 55.00000000 30.00000000,
   55.00000000 40.00000000, 54.00000000 45.00000000, 45.00000000
   50.00000000, 30.00000000 50.00000000, 25.00000000 49.00000000,
   20.00000000 40.00000000, 20.00000000 30.00000000, 21.00000000
   25.00000000, 30.00000000 20.00000000, 45.00000000 20.00000000,
   50.00000000 21.00000000, 55.00000000 30.00000000))
```

Exemplo 2:

A instrução SELECT a seguir utiliza a função ST_Buffer para aplicar um buffer negativo de 5.

```
SELECT id, spatial_type,
       cast(ST_AsText(ST_Buffer(geometry, -5)) AS varchar(150))
       AS buffer_negative_5
FROM   sample_geometries
WHERE  id = 3
```

Resultados:

```
ID          SPATIAL_TYPE       BUFFER_NEGATIVE_5
----------- ------------------ ------------------------------------
3           st_polygon         POLYGON (( 115.00000000 125.00000000,
   125.00000000 125.00000000, 125.00000000 135.00000000, 115.00000000
   135.00000000, 115.00000000 125.00000000))
```

Exemplo 3:

A instrução SELECT a seguir mostra o resultado da aplicação de um buffer com o parâmetro unit especificado.

```
SELECT id, spatial_type,
    cast(ST_AsText(ST_Buffer(geometry, 10, 'METER')) AS varchar(680))
    AS buffer_10_meter
FROM   sample_geometries
WHERE  id = 3
```

Resultados:

```
ID          SPATIAL_TYPE       BUFFER_10_METER
----------- ------------------ ------------------------------------
3           st_polygon         POLYGON (( 163.00000000 120.00000000,
   163.00000000 140.00000000, 162.00000000 149.00000000, 159.00000000
   157.00000000, 152.00000000 165.00000000, 143.00000000 170.00000000,
   130.00000000 173.00000000, 110.00000000 173.00000000, 101.00000000
   172.00000000, 92.00000000 167.00000000, 84.00000000 160.00000000,
   79.00000000 151.00000000, 77.00000000 140.00000000, 77.00000000
   120.00000000, 78.00000000 111.00000000, 83.00000000 102.00000000,
   90.00000000 94.00000000, 99.00000000 89.00000000, 110.00000000
   87.00000000, 130.00000000 87.00000000, 139.00000000 88.00000000,
   147.00000000 91.00000000, 155.00000000 98.00000000, 160.00000000
   107.00000000, 163.00000000 120.00000000))
```

Referência Relacionada:

- "Funções Espaciais Suportadas pelo DB2 Geodetic Extender" na página 208
- "A Exibição do Catálogo DB2GSE.ST_UNITS_OF_MEASURE" na página 298

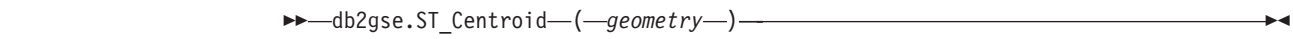
ST_Centroid

ST_Centroid utiliza uma geometria como um parâmetro de entrada e retorna o centro geométrico, que é o centro do retângulo limitador mínimo da geometria especificada, como um ponto. O ponto resultante é representado no sistema de referência espacial da geometria especificada.

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria para determinar o centro geométrico.

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

Este exemplo cria duas geometrias e encontra o centróide delas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
    (1, ST_Polygon('polygon
    ((40 120, 90 120, 90 150, 40 150, 40 120),
    (50 130, 80 130, 80 140, 50 140, 50 130))',0))

INSERT INTO sample_geoms VALUES
    (2, ST_MultiPoint('multipoint(10 10, 50 10, 10 30)' ,0))


SELECT id, CAST(ST_AsText(ST_Centroid(geometry))
   as VARCHAR(40)) Centroid
FROM sample_geoms
```

Resultados:

```
ID          CENTROID
----------- ----------------------------------------
          1 POINT ( 65.00000000 135.00000000)
          2 POINT ( 30.00000000 20.00000000)
```

ST_ChangePoint

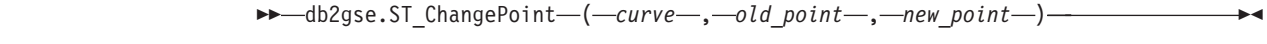
ST_ChangePoint utiliza uma curva e dois pontos como parâmetros de entrada. Substitui todas as ocorrências do primeiro ponto na curva especificada pelo segundo ponto e retorna a curva resultante. A geometria resultante é representada no sistema de referência espacial da geometria especificada.

Se os dois pontos não forem representados no mesmo sistema de referência espacial como a curva, eles serão convertidos para o sistema de referência espacial utilizado para a curva.

Se a curva for vazia, então um valor vazio será retornado. Se a curva especificada for nula ou se qualquer um dos pontos fornecidos for nulo ou vazio, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

curve Um valor do tipo ST_Curve ou um de seus subtipos que representa a curva em que os pontos identificados por *old_point* são alterados para *new_point*.

old_point
Um valor do tipo ST_Point que identifica os pontos na curva que são alterados para *new_point*.

new_point
Um valor do tipo ST_Point que representa as novas localizações dos pontos na curva identificada por *old_point*.

Tipo de Retorno:

db2gse.ST_Curve

Restrições:

O ponto a ser alterado na curva deve ser um dos pontos utilizados para definir a curva.

Se a curva tiver coordenadas Z ou M, os pontos especificados também deverão ter coordenadas Z ou M.

Exemplos:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir cria e ocupa a tabela SAMPLE_LINES.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines(id INTEGER, line ST_Linestring)


INSERT INTO sample_lines VALUES
    (1, ST_LineString('linestring (10 10, 5 5, 0 0, 10 0, 5 5, 0 10)', 0) )

INSERT INTO sample_lines VALUES
    (2, ST_LineString('linestring z (0 0 4, 5 5 5, 10 10 6, 5 5 7)', 0) )
```

Exemplo 1:

Este exemplo altera todas as ocorrências do ponto (5, 5) para o ponto (6, 6) na cadeia de linhas.

```
SELECT cast(ST_AsText(ST_ChangePoint(line, ST_Point(5, 5),
                                     ST_Point(6, 6))) as VARCHAR(160))
FROM   sample_lines
WHERE  id=1
```

Resultados:

```
NEW
--------------------------------------------------------------------------
LINESTRING ( 10.00000000 10.00000000, 6.00000000 6.00000000, 0.00000000
0.00000000, 10.00000000 0.00000000, 6.00000000 6.00000000, 0.00000000
10.00000000)
```

Exemplo 2:

Este exemplo altera todas as ocorrências do ponto (5, 5, 5) para o ponto (6, 6, 6) na cadeia de linhas.

```
SELECT cast(ST_AsText(ST_ChangePoint(line, ST_Point(5.0, 5.0, 5.0),
       ST_Point(6.0, 6.0, 6.0) )) as VARCHAR(180))
FROM   sample_lines
WHERE  id=2
```

Resultados:

```
NEW
-----------------------------------------------------------------------------
LINESTRING Z ( 0.00000000 0.00000000 4.00000000, 6.00000000 6.00000000
6.00000000, 10.00000000 10.00000000 6.00000000, 5.00000000 5.00000000
7.00000000)
```

ST_Contains

ST_Contains utiliza duas geometrias como parâmetros de entrada e retorna 1 se a primeira geometria contiver totalmente a segunda; caso contrário, retorna 0 (zero) para indicar que a primeira geometria não contém totalmente a segunda.

Se qualquer uma das geometrias especificadas for nula ou vazia, será retornado nulo.

Para dados não geodésicos, se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial. Para dados geodésicos, as duas geometrias devem estar no mesmo sistema de referência espacial (SRS) geodésico.

Sintaxe:

Parâmetro:

geometry1
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que deve ser testada para conter totalmente *geometry2*.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que deve ser testada para que esteja totalmente dentro de *geometry1*.

Restrições: Para dados geodésicos, as duas geometrias devem ser geodésicas e devem estar no mesmo SRS geodésico.

Tipo de Retorno:

INTEGER

Exemplos:

O código a seguir cria e ocupa estas tabelas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_points(id SMALLINT, geometry ST_POINT)

CREATE TABLE sample_lines(id SMALLINT, geometry ST_LINESTRING)

CREATE TABLE sample_polygons(id SMALLINT, geometry ST_POLYGON)

INSERT INTO sample_points (id, geometry)
VALUES
    (1, ST_Point(10, 20, 1)),
    (2, ST_Point('point(41 41)', 1))

INSERT INTO sample_lines (id, geometry)
VALUES
    (10, ST_LineString('linestring (1 10, 3 12, 10 10)', 1) ),
    (20, ST_LineString('linestring (50 10, 50 12, 45 10)', 1) )
INSERT INTO sample_polygons(id, geometry)
VALUES
      (100, ST_Polygon('polygon((0 0, 0 40, 40 40, 40 0, 0 0))',  1) )
```

Exemplo 1:

O fragmento de código a seguir utiliza a função ST_Contains para determinar quais pontos são contidos por um polígono específico.

```
SELECT poly.id AS polygon_id,
       CASE ST_Contains(poly.geometry, pts.geometry)
          WHEN 0 THEN 'does not contain'
          WHEN 1 THEN 'does contain'
       END AS contains,
       pts.id AS point_id
FROM   sample_points pts, sample_polygons poly
```

Resultados:

```
POLYGON_ID CONTAINS         POINT_ID
---------- ---------------- --------
       100 does contain            1
       100 does not contain        2
```

Exemplo 2:

O fragmento de código a seguir utiliza a função ST_Contains para determinar quais linhas são contidas por um polígono específico.

```
SELECT poly.id AS polygon_id,
       CASE ST_Contains(poly.geometry, line.geometry)
          WHEN 0 THEN 'does not contain'
          WHEN 1 THEN 'does contain
       END AS contains,
       line.id AS line_id
FROM   sample_lines line, sample_polygons poly
```

Resultados:

```
POLYGON_ID CONTAINS         LINE_ID
---------- ---------------- -------
       100 does contain          10
       100 does not contain      20
```

Referência Relacionada:

- "ST_Within" na página 502

ST_ConvexHull

ST_ConvexHull utiliza uma geometria como um parâmetro de entrada e retorna o envoltório convexo dele.

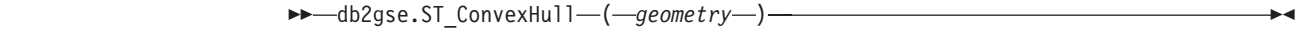
A geometria resultante é representada no sistema de referência espacial da geometria especificada.

Se possível, o tipo específico da geometria retornada será ST_Point, ST_LineString ou ST_Polygon. Por exemplo, o limite de um polígono sem aberturas é uma cadeia de linhas única, representada como ST_LineString. O limite de um polígono com uma ou mais aberturas consiste em várias cadeias de linhas, representadas como ST_MultiLineString.

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria para calcular o envoltório convexo.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir cria e ocupa a tabela SAMPLE_GEOMETRIES.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries(id INTEGER, spatial_type varchar(18),
   geometry ST_GEOMETRY)

INSERT INTO sample_geometries(id, spatial_type, geometry)
VALUES
    (1, 'ST_LineString', ST_LineString
        ('linestring(20 20, 30 30, 20 40, 30 50)', 0)),
```

```
    (2, 'ST_Polygon', ST_Polygon('polygon
        ((110  120,  110 140, 120 130, 110 120))', 0) ),
    (3, 'ST_Polygon', ST_Polygon('polygon((30 30, 25 35, 15 50,
        35 80, 40 85, 80 90,70 75, 65 70, 55 50, 75 40, 60 30,
        30 30))', 0) ),
    (4, 'ST_MultiPoint', ST_MultiPoint('multipoint(20 20, 30 30,
        20 40, 30 50)', 1))
```

A instrução SELECT a seguir calcula o envoltório convexo para todas as geometrias construídas acima e exibe o resultado.

```
SELECT id, spatial_type, cast(geometry..ST_ConvexHull..ST_AsText
   AS varchar(300)) AS convexhull
FROM   sample_geometries
```

Resultados:

```
ID     SPATIAL_TYPE       CONVEXHULL
------ ------------------ --------------------------------------------------
     1 ST_LineString      POLYGON (( 20.00000000 40.00000000,
                            20.00000000 20.00000000, 30.00000000
                            30.00000000, 30.00000000 50.00000000,
                            20.00000000 40.00000000))

     2 ST_Polygon         POLYGON (( 110.00000000 140.00000000,
                            110.00000000 120.00000000, 120.00000000
                            130.00000000,  110.00000000 140.00000000))

     3 ST_Polygon         POLYGON (( 15.00000000 50.00000000,
                            25.00000000 35.00000000, 30.00000000
                            30.00000000, 60.00000000 30.00000000,
                            75.00000000 40.00000000, 80.00000000
                            90.00000000, 40.00000000 85.00000000,
                            35.00000000 80.00000000, 15.00000000
                            50.00000000))

     4 ST_MultiPoint      POLYGON (( 20.00000000 40.00000000,
                            20.00000000 20.00000000, 30.00000000
                            30.00000000, 30.00000000 50.00000000,
                            20.00000000 40.00000000))
```

ST_CoordDim

ST_CoordDim utiliza uma geometria como parâmetro de entrada e retorna a dimensão de suas coordenadas.

Se a geometria especificada não tiver coordenadas Z e M, a dimensão será 2. Se tiver coordenadas Z e nenhuma coordenada M, ou se tiver coordenadas M e nenhuma coordenada Z, a dimensão será 3. Se tiver coordenadas Z e M, a dimensão será 4. Se a geometria for nula, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria a partir da qual a dimensão será recuperada.

Tipo de Retorno:

INTEGER

Exemplo:

Este exemplo cria várias geometrias e depois determina a dimensão de suas coordenadas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id CHARACTER(15), geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       ('Empty Point', ST_Geometry('point EMPTY',0))

INSERT INTO sample_geoms VALUES
       ('Linestring', ST_Geometry('linestring (10 10, 15 20)',0))

INSERT INTO sample_geoms VALUES
       ('Polygon', ST_Geometry('polygon((40 120, 90 120, 90 150,
         40 150, 40 120))' ,0))

INSERT INTO sample_geoms VALUES
       ('Multipoint M', ST_Geometry('multipoint m (10 10 5, 50 10
         6, 10 30 8)' ,0))

INSERT INTO sample_geoms VALUES
       ('Multipoint Z', ST_Geometry('multipoint z (47 34 295,
         23 45 678)' ,0))

INSERT INTO sample_geoms VALUES
       ('Point ZM',     ST_Geometry('point zm (10 10 16 30)' ,0))


SELECT id, ST_CoordDim(geometry)  COORDDIM
FROM sample_geoms
```

Resultados:

```
ID              COORDDIM
--------------- -----------
Empty Point               2
Linestring                2
Polygon                   2
Multipoint M              3
Multipoint Z              3
Point ZM                  4
```

ST_Crosses

ST_Crosses utiliza duas geometrias como parâmetros de entrada e retorna 1 se a primeira geometria cruzar com a segunda. Caso contrário, será retornado 0 (zero).

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Se a primeira geometria for um polígono ou um multipolígono, ou se a segunda geometria for um ponto ou um multiponto ou se qualquer uma das geometrias for um valor nulo ou estiver vazia, será retornado nulo. Se a interseção das duas geometrias resultar em uma geometria que tenha uma dimensão menor do que a

dimensão máxima das duas geometrias especificadas e se a geometria resultante não for igual à nenhuma das duas geometrias especificadas, será retornado 1. Caso contrário, o resultado será 0 (zero).

Sintaxe:

Parâmetro:

geometry1
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que deve ser testada para cruzar *geometry2*.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que deve ser testada para determinar se ela é cruzada por *geometry1*.

Tipo de Retorno:

INTEGER

Exemplo:

Este código determina se as geometrias construídas se cruzam.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('polygon((30 30, 30 50, 50 50, 50 30, 30 30))' ,0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('linestring(40 50, 50 40)' ,0))

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('linestring(20 20, 60 60)' ,0))

SELECT a.id, b.id, ST_Crosses(a.geometry, b.geometry) Crosses
FROM sample_geoms a, sample_geoms b
```

Resultados:

```
ID          ID          CROSSES
----------- ----------- -----------
          1           1           -
          2           1           0
          3           1           1
          1           2           -
          2           2           0
          3           2           1
          1           3           -
          2           3           1
          3           3           0
```

Referência Relacionada:

- "Funções que Comparam Recursos Geográficos" na página 308

ST_Difference

ST_Difference utiliza duas geometrias como parâmetros de entrada e retorna a parte da primeira geometria que não cruza com a segunda geometria.

As duas geometrias devem ter a mesma dimensão. Se qualquer uma das geometrias for nula, será retornado nulo. Se a primeira geometria for vazia, será retornada uma geometria vazia do tipo ST_Point. Se a segunda geometria for vazia, será retornada a primeira geometria inalterada.

Para dados não geodésicos, se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial. Para dados geodésicos, as duas geometrias devem estar no mesmo SRS (Spatial Reference System) geodésico.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry1
: Um valor do tipo ST_Geometry que representa a primeira geometria a ser utilizada para calcular a diferença para *geometry2*.

geometry2
: Um valor do tipo ST_Geometry que representa a segunda geometria que é utilizada para calcular a diferença para *geometry1*.

Restrições para dados geodésicos:

- As duas geometrias devem ser geodésicas e devem estar no mesmo SRS geodésico.
- ST_Difference suporta apenas os tipos de dados ST_Point, ST_Polygon, ST_MultiPoint e ST_MultiPolygon.

Tipo de Retorno:

db2gse.ST_Geometry

A dimensão da geometria retornada é igual à das geometrias de entrada.

Exemplos:

No exemplo a seguir, os resultados foram reformatados para possibilitar a leitura. O espaçamento nos resultados irá variar de acordo com a exibição.

O código a seguir cria e ocupa a tabela SAMPLE_GEOMETRIES.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('polygon((10 10, 10 20, 20 20, 20 10, 10 10))' ,0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('polygon((30 30, 30 50, 50 50, 50 30, 30 30))' ,0))
```

```
INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('polygon((40 40, 40 60, 60 60, 60 40, 40 40))' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('linestring(70 70, 80 80)' ,0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('linestring(75 75, 90 90)' ,0))
```

Exemplo 1:

Este exemplo encontra a diferença entre dois polígonos separados.

```
SELECT a.id, b.id, CAST(ST_AsText(ST_Difference(a.geometry, b.geometry))
   as VARCHAR(200))  Difference
FROM sample_geoms a, sample_geoms b
WHERE a.id = 1 and b.id = 2
```

Resultados:

```
ID       ID       DIFFERENCE
-------- -------- ----------------------------------------------------
       1        2 POLYGON (( 10.00000000 10.00000000, 20.00000000
                     10.00000000, 20.00000000 20.00000000,
                     10.00000000 20.00000000, 10.00000000 10.00000000))
```

Exemplo 2:

Este exemplo encontra a diferença entre dois polígonos que fazem interseção.

```
SELECT a.id, b.id, CAST(ST_AsText(ST_Difference(a.geometry, b.geometry))
   as VARCHAR(200))  Difference
FROM sample_geoms a, sample_geoms b
WHERE a.id = 2 and b.id = 3
```

Resultados:

```
ID       ID       DIFFERENCE
-------- -------- ----------------------------------------------------
       2        3 POLYGON (( 30.00000000 30.00000000, 50.00000000
                     30.00000000, 50.00000000 40.00000000, 40.00000000
                     40.00000000, 40.00000000 50.00000000, 30.00000000
                     50.00000000, 30.00000000 30.00000000))
```

Exemplo 3:

Este exemplo encontra a diferença entre duas cadeias de linhas de sobreposição.

```
SELECT a.id, b.id, CAST(ST_AsText(ST_Difference(a.geometry, b.geometry))
   as VARCHAR(100))  Difference
FROM sample_geoms a, sample_geoms b
WHERE a.id = 4 and b.id = 5
```

Resultados:

```
ID       ID       DIFFERENCE
-------- -------- ----------------------------------------------------
       4        5 LINESTRING ( 70.00000000 70.00000000, 75.00000000
                     75.00000000)
```

ST_Dimension

ST_Dimension utiliza uma geometria como um parâmetro de entrada e retorna sua dimensão.

Se a geometria especificada for vazia, -1 será retornado. Para pontos e multipontos, a dimensão é 0 (zero); para curvas e multicurvas, a dimensão é 1; e para polígonos e multipolígonos, a dimensão é 2. Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry que representa a geometria para a qual a dimensão é retornada.

Tipo de Retorno:

INTEGER

Exemplo:

Este exemplo cria várias geometrias diferentes e encontra suas dimensões.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id char(15), geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       ('Empty Point', ST_Geometry('point EMPTY',0))

INSERT INTO sample_geoms VALUES
       ('Point ZM', ST_Geometry('point zm (10 10 16 30)' ,0))

INSERT INTO sample_geoms VALUES
       ('MultiPoint M', ST_Geometry('multipoint m (10 10 5,
         50 10 6, 10 30 8)' ,0))

INSERT INTO sample_geoms VALUES
       ('LineString', ST_Geometry('linestring (10 10, 15 20)',0))

INSERT INTO sample_geoms VALUES
       ('Polygon', ST_Geometry('polygon((40 120, 90 120, 90 150,
         40 150, 40 120))' ,0))

SELECT id, ST_Dimension(geometry)  Dimension
FROM sample_geoms
```

Resultados:

```
ID              DIMENSION
--------------- -----------
Empty Point              -1
Point ZM                  0
MultiPoint M              0
LineString                1
Polygon                   2
```

ST_Disjoint

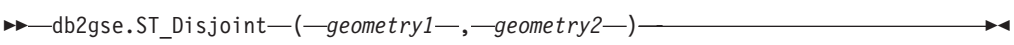
ST_Disjoint utiliza duas geometrias como parâmetros de entrada e retorna 1 se as geometrias especificadas não forem interseccionadas. Se as geometrias forem interseccionadas, será retornado 0 (zero).

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Se qualquer uma das duas geometrias for nula ou vazia, será retornado um valor nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry1
Um valor do tipo ST_Geometry que representa a geometria que é testada para ser separada de *geometry2*.

geometry2
Um valor do tipo ST_Geometry que representa a geometria que será testada para ser separada de *geometry1*.

Tipo de Retorno:

INTEGER

Exemplos:

Este código cria várias geometrias na tabela SAMPLE_GEOMETRIES.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('polygon((20 30, 30 30, 30 40, 20 40, 20 30))',0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('polygon((30 30, 30 50, 50 50, 50 30, 30 30))',0))

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('polygon((40 40, 40 60, 60 60, 60 40, 40 40))',0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('linestring(60 60, 70 70)' ,0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('linestring(30 30, 40 40)' ,0))
```

Exemplo 1:

Este exemplo determina se o primeiro polígono está separado de qualquer uma das geometrias.

```
SELECT a.id, b.id, ST_Disjoint(a.geometry, b.geometry) DisJoint
FROM sample_geoms a, sample_geoms b
WHERE a.id = 1
```

Resultados:

```
ID          ID          DISJOINT
----------- ----------- -----------
          1           1           0
          1           2           0
          1           3           1
          1           4           1
          1           5           0
```

Exemplo 2:

Este exemplo determina se o terceiro polígono está separado de qualquer uma das geometrias.

```
SELECT a.id, b.id, ST_Disjoint(a.geometry, b.geometry) DisJoint
FROM sample_geoms a, sample_geoms b
WHERE a.id = 3
```

Resultados:

```
ID          ID          DISJOINT
----------- ----------- -----------
          3           1           1
          3           2           0
          3           3           0
          3           4           0
          3           5           0
```

Exemplo 3:

Este exemplo determina se a segunda cadeia de linhas está separada de qualquer uma das geometrias.

```
SELECT a.id, b.id, ST_Disjoint(a.geometry, b.geometry) DisJoint
FROM sample_geoms a, sample_geoms b
WHERE a.id = 5
```

Resultados:

```
ID          ID          DISJOINT
----------- ----------- -----------
          5           1           0
          5           2           0
          5           3           0
          5           4           1
          5           5           0
```

Referência Relacionada:

• "Funções que Comparam Recursos Geográficos" na página 308

ST_Distance

ST_Distance utiliza duas geometrias e, opcionalmente, uma unidade como parâmetros de entrada e retorna a distância mais curta entre qualquer ponto da primeira geometria até qualquer ponto da segunda geometria, medido nas unidades padrão ou fornecidas.

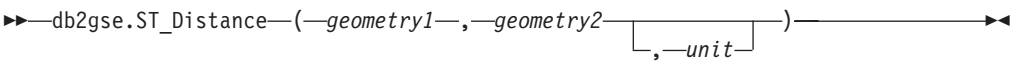
Para dados geodésicos, ST_Distance retorna a *distância geodésica* entre duas geometrias. A distância geodésica é a menor distância na superfície do elipsóide. Para obter informações adicionais, consulte "Distâncias Geodésicas" na página 162.

Se qualquer uma das duas geometrias for nula ou vazia, será retornado nulo.

Para dados não geodésicos, se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial. Para dados geodésicos, as duas geometrias devem estar no mesmo sistema de referência espacial (SRS) geodésico.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry1
Um valor do tipo ST_Geometry que representa a geometria que é utilizada para calcular a distância para *geometry2*.

geometry2
Um valor do tipo ST_Geometry que representa a geometria que é utilizada para calcular a distância para *geometry1*.

unit VARCHAR(128) valor que identifica a unidade na qual o resultado é medido. As unidades de medida suportadas estão listadas na exibição do catálogo DB2GSE.ST_UNITS_OF_MEASURE.

Para dados geodésicos, as duas geometrias devem ser geodésicas e devem estar no mesmo SRS geodésico.

Se o parâmetro *unit* for omitido, as seguintes regras serão utilizadas para determinar a unidade de medida utilizada para o resultado:

- Se *geometry1* estiver em um sistema de coordenadas projetadas ou geocêntricas, a unidade linear associada a este sistema de coordenadas será a padrão.
- Se *geometry1* estiver em um sistema de coordenadas geográficas, mas não estiver em um SRS geodésico, a unidade angular associada a este sistema de coordenadas será a padrão.
- Se *geometry1* estiver em um SRS geodésico, a unidade de medida padrão será metros.

Restrições para conversões de unidades: Será retornado um erro (SQLSTATE 38SU4) se ocorrer qualquer uma das condições a seguir:

- A geometria está em um sistema de coordenadas não especificado e o parâmetro *unit* é especificado.
- A geometria está em um sistema de coordenadas projetadas e uma unidade angular é especificada.
- A geometria está em um sistema de coordenadas geográficas, mas não está em um SRS geodésico e uma unidade linear é especificada.
- A geometria está em um sistema de coordenadas geográficas, está em um SRS geodésico e uma unidade angular é especificada.

Tipo de Retorno:

DOUBLE

Exemplos:

As instruções SQL a seguir criam e ocupam as tabelas SAMPLE_GEOMETRIES1 e SAMPLE_GEOMETRIES2.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries1(id SMALLINT, spatial_type varchar(13),
   geometry ST_GEOMETRY)

CREATE TABLE sample_geometries2(id SMALLINT, spatial_type varchar(13),
   geometry ST_GEOMETRY)

INSERT INTO sample_geometries1(id, spatial_type, geometry)
VALUES
    ( 1, 'ST_Point', ST_Point('point(100 100)', 1) ),
    (10, 'ST_LineString', ST_LineString('linestring(125 125, 125 175)', 1) ),
    (20, 'ST_Polygon', ST_Polygon('polygon
                        ((50 50, 50 150, 150 150, 150 50, 50 50))', 1) )

INSERT INTO sample_geometries2(id, spatial_type, geometry)
VALUES
    (101, 'ST_Point', ST_Point('point(200 200)', 1) ),
    (102, 'ST_Point', ST_Point('point(200 300)', 1) ),
    (103, 'ST_Point', ST_Point('point(200 0)', 1) ),
    (110, 'ST_LineString', ST_LineString('linestring(200 100, 200 200)', 1) ),
    (120, 'ST_Polygon', ST_Polygon('polygon
                         ((200 0, 200 200, 300 200, 300 0, 200 0))', 1) )
```

Exemplo 1:

A seguinte instrução SELECT calcula a distância entre as diversas geometrias nas tabelas SAMPLE_GEOMTRIES1 e SAMPLE_GEOMTRIES2.

```
SELECT   sg1.id AS sg1_id, sg1.spatial_type AS sg1_type,
         sg2.id AS sg1_id, sg2.spatial_type AS sg2_type,
         cast(ST_Distance(sg1.geometry, sg2.geometry)
         AS Decimal(8, 4)) AS distance
FROM     sample_geometries1 sg1, sample_geometries2 sg2
ORDER BY sg1.id
```

Resultados:

```
SG1_ID SG1_TYPE      SG1_ID SG2_TYPE      DISTANCE
------ ------------- ------ ------------- ----------
     1 ST_Point         101 ST_Point        141.4213
     1 ST_Point         102 ST_Point        223.6067
     1 ST_Point         103 ST_Point        141.4213
     1 ST_Point         110 ST_LineString   100.0000
     1 ST_Point         120 ST_Polygon      100.0000
    10 ST_LineString    101 ST_Point         79.0569
    10 ST_LineString    102 ST_Point        145.7737
    10 ST_LineString    103 ST_Point        145.7737
    10 ST_LineString    110 ST_LineString    75.0000
    10 ST_LineString    120 ST_Polygon       75.0000
    20 ST_Polygon       101 ST_Point         70.7106
    20 ST_Polygon       102 ST_Point        158.1138
    20 ST_Polygon       103 ST_Point         70.7106
    20 ST_Polygon       110 ST_LineString    50.0000
    20 ST_Polygon       120 ST_Polygon       50.0000
```

Exemplo 2:

A seguinte instrução SELECT ilustra como encontrar todas as geometrias que estão a uma distância de 100 umas das outras.

```
SELECT   sg1.id AS sg1_id, sg1.spatial_type AS sg1_type,
         sg2.id AS sg1_id, sg2.spatial_type AS sg2_type,
         cast(ST_Distance(sg1.geometry, sg2.geometry)
         AS Decimal(8, 4)) AS distance
FROM     sample_geometries1 sg1, sample_geometries2 sg2
WHERE    ST_Distance(sg1.geometry, sg2.geometry)  <= 100
```

Resultados:

```
SG1_ID SG1_TYPE      SG1_ID SG2_TYPE      DISTANCE
------ ------------- ------ ------------- ----------
     1 ST_Point         110 ST_LineString   100.0000
     1 ST_Point         120 ST_Polygon      100.0000
    10 ST_LineString    101 ST_Point         79.0569
    10 ST_LineString    110 ST_LineString    75.0000
    10 ST_LineString    120 ST_Polygon       75.0000
    20 ST_Polygon       101 ST_Point         70.7106
    20 ST_Polygon       103 ST_Point         70.7106
    20 ST_Polygon       110 ST_LineString    50.0000
    20 ST_Polygon       120 ST_Polygon       50.0000
```

Exemplo 3:

A seguinte instrução SELECT calcula a distância, em quilômetros, entre as diversas geometrias.

```
Tabelas SAMPLE_GEOMTRIES1 e SAMPLE_GEOMTRIES2.
SELECT   sg1.id AS sg1_id, sg1.spatial_type AS sg1_type,
         sg2.id AS sg1_id, sg2.spatial_type AS sg2_type,
         cast(ST_Distance(sg1.geometry, sg2.geometry, 'KILOMETER')
         AS DECIMAL(10, 4)) AS distance
FROM     sample_geometries1 sg1, sample_geometries2 sg2
ORDER BY sg1.id
```

Resultados:

```
SG1_ID SG1_TYPE      SG1_ID SG2_TYPE      DISTANCE
------ ------------- ------ ------------- ------------
     1 ST_Point         101 ST_Point        12373.2168
     1 ST_Point         102 ST_Point        16311.3816
     1 ST_Point         103 ST_Point         9809.4713
     1 ST_Point         110 ST_LineString    1707.4463
     1 ST_Point         120 ST_Polygon      12373.2168
    10 ST_LineString    101 ST_Point         8648.2333
    10 ST_LineString    102 ST_Point        11317.3934
    10 ST_LineString    103 ST_Point        10959.7313
    10 ST_LineString    110 ST_LineString    3753.5862
    10 ST_LineString    120 ST_Polygon      10891.1254
    20 ST_Polygon       101 ST_Point         7700.5333
    20 ST_Polygon       102 ST_Point        15039.8109
    20 ST_Polygon       103 ST_Point         7284.8552
    20 ST_Polygon       110 ST_LineString    6001.8407
    20 ST_Polygon       120 ST_Polygon      14515.8872
```

Referência Relacionada:

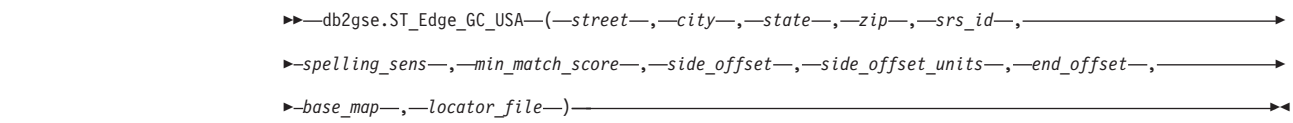
- "Funções que Comparam Recursos Geográficos" na página 308

ST_Edge_GC_USA

ST_Edge_GC_USA é a função que implementa DB2SE_USA_GEOCODER que executa geocode de endereços localizados nos Estados Unidos da América em pontos. Os endereços são comparados (correspondidos) com arquivos EDGE, que são fornecidos no CD de dados do geocoder.

A função utiliza o número e nome da rua, o nome da cidade, o estado, o CEP e o identificador do sistema de referência espacial para o ponto resultante como parâmetros de entrada e retorna um valor ST_Point. Além disso, vários parâmetros de configuração que influenciam o processo de geocoding podem ser especificados.

Sintaxe:



Parâmetro:

street Um valor do tipo VARCHAR(128) que contém o número e nome da rua do endereço do qual será executado geocode.

Este valor não deve ser nulo.

city Um valor do tipo VARCHAR(128) que contém o nome da cidade do endereço do qual será executado geocode.

Este valor pode ser nulo se o parâmetro *zip* for especificado.

state Um valor do tipo VARCHAR(128) que contém o nome do estado do endereço do qual será executado geocode. O estado pode ser abreviado ou não.

Este valor pode ser nulo se o parâmetro *zip* for especificado.

zip Um valor do tipo VARCHAR(10) que contém o CEP do endereço do qual será executado geocode. O CEP pode ter 5 dígitos ou estar em notação 5+4.

Este valor pode ser nulo se os parâmetros *city* e *state* forem especificados.

srs_id Um valor do tipo INTEGER que contém o identificador numérico do sistema de referência espacial do ponto resultante. O valor deve identificar um sistema de referência espacial existente, que utiliza um sistema de coordenadas projetadas, com base no sistema de coordenadas geográficas GCS_NORTH_AMERICAN_1983, ou em um sistema de referência espacial existente que utiliza o próprio sistema de coordenadas geográficas, GCS_NORTH_AMERICAN_1983.

Se *srs_id* não identifica um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então um erro é retornado (SQLSTATE 38SU1).

spelling_sens
Um valor do tipo INTEGER que especifica a distinção entre maiúsculas e minúsculas que deve ser aplicada ao endereço especificado. O valor deve estar no intervalo de 0 (zero) a 100. Quanto mais alto o valor, mais limitado será o geocoder com relação às diferenças na ortografia do endereço especificado. Os desvios resultam em uma maior penalidade que será aplicada no score final da correspondência.

Se distinção entre maiúsculas e minúsculas for definida como muito alta, poucos endereços terão o geocode executado com êxito e será retornado um valor nulo. Se a distinção entre maiúsculas e minúsculas for definida como muito baixa, um número maior de endereços não correspondentes poderá ser considerado como correspondências corretas, devido ao nível de diferença aceito na ortografia dos endereços. **Recomendação:** Defina este valor como 60.

Se este valor for nulo, a distinção entre maiúsculas e minúsculas será derivada do arquivo localizador. Se ele não for especificado no arquivo localizador, será utilizada a distinção entre maiúsculas e minúsculas com valor 60.

min_match_score
Um valor do tipo INTEGER que contém o valor de score mínimo que um ponto deve ter para ser considerado uma correspondência para o endereço especificado. O valor de score mínimo deve estar no intervalo de 0 (zero) a 100. Se o score do ponto for menor do que o valor *min_match_score*, será retornado nulo em vez do ponto e não será efetuado geocode do endereço.

Diferentes fatores como a qualidade do mapa base, a distinção entre maiúsculas e minúsculas ou a precisão se o endereço influenciar o score de um ponto. **Recomendação:** Defina este valor como 80.

Se este valor for nulo, o score mínimo de correspondência será derivado do arquivo localizador. Se ele não for especificado no arquivo localizador, será utilizado um valor de score mínimo de 80.

side_offset
Um valor do tipo DOUBLE que especifica a distância que um ponto resultante deve ser colocado do centro da rua. O valor deve ser maior ou igual a 0 (zero). O parâmetro *side_offset_unit* identifica as unidades que são utilizadas para medir o deslocamento lateral.

Se este valor for nulo, o deslocamento lateral será derivado do arquivo localizador. Se ele não for especificado no arquivo localizador, será utilizado um deslocamento lateral de 0.0.

side_offset_units
Um valor do tipo VARCHAR(128) que contém as unidades na qual o parâmetro *side_offset* é medido. O valor deve ser uma das seguintes unidades:

- Polegadas
- Pontos
- Pés
- Jardas
- Milhas
- Milhas náuticas
- Milímetros
- Centímetros
- Metros
- Quilômetros
- Graus decimais
- Metros projetados
- Unidades de dados de referência

Se este valor for nulo, as unidades de deslocamento lateral serão derivadas do arquivo localizador. Se ele não for especificado no arquivo localizador, o deslocamento lateral será medido em pés.

end_offset
Um valor do tipo INTEGER que indica a que distância um ponto que está exatamente no final de um segmento de rua deve ser colocado no segmento. O valor deve ser maior ou igual a 0 (zero). Este parâmetro é utilizado para evitar a colocação de pontos resultantes no meio de uma rua em interseções. O deslocamento final é medido em pontos (a menor resolução possível) no mapa base.

Se este valor for nulo, o deslocamento final será derivado do arquivo localizador. Se ele não for especificado no arquivo localizador, será utilizado um deslocamento final de 3.

base_map
Um valor do tipo VARCHAR(256) que contém o caminho completo, incluindo o nome base, para o arquivo de mapa base (.edg). O arquivo de mapa base é utilizado pelo geocoder para corresponder os endereços especificados. Os mapas base fornecidos pelo DB2 Spatial Extender devem ser utilizados. Você pode utilizar este parâmetro se tiver colocado os mapas base em um diretório diferente.

Se este valor for nulo, o caminho para o mapa base será derivado do arquivo localizador. Se ele não for especificado no arquivo localizador, o mapa base será procurado no diretório sqllib da instância atual, no subdiretório gse/refdata. O nome base do arquivo procurado é usa.edg.

locator_file
Um valor do tipo VARCHAR(256) que contém o caminho completo, incluindo o nome base, para o arquivo localizador que contém parâmetros de configuração adicionais para o geocoder. O arquivo localizador fornecido pelo DB2 Spatial Extender deve ser utilizado.

Se este valor for nulo, o arquivo localizador será procurado no diretório sqllib da instância atual, no subdiretório gse/cfg/geocoder. O nome base do arquivo pesquisado é EDGELocator.loc.

Tipo de Retorno:

db2gse.ST_Point

Exemplos:

Exemplo 1:

O código a seguir cria uma tabela SAMPLE_GEOCODING e insere dois endereços dos quais é efetuado geocode subseqüentemente. O score mínimo de correspondência será definido como 50 para os endereços especificados, e o sistema de referência espacial para os pontos resultantes será 1.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geocoding (
 street VARCHAR(128),
  city   VARCHAR(128),
 state  VARCHAR(128),
 zip    VARCHAR(5) )

INSERT INTO geocoding(street, city, state, zip)
```

```
 VALUES ('1212 New York Ave NW', 'Washington', 'DC', '20005'),
 ('100 First North Street', 'San Jose', 'CA', NULL)

SELECT VARCHAR(ST_AsText(ST_Edge_GC_USA(street, city, state, zip, 1,
    CAST(NULL AS INTEGER), 50, CAST(NULL AS DOUBLE),
    CAST(NULL AS VARCHAR(128)), CAST(NULL AS INTEGER),
     CAST(NULL AS VARCHAR(256)), CAST(NULL AS VARCHAR(256)))), 50)
FROM sample_geocoding
```

Resultados:

```
1
--------------------------------------------------
POINT ( -77.02829300 38.90049000)
POINT ( -121.94507200 37.28766700)
```

Exemplo 2:

Neste exemplo, é criado um sistema de referência espacial que utiliza um sistema de coordenadas projetadas. Para simplificar a interface da função de geocoding, é criada uma função definida pelo usuário para agrupar a função ST_Edge_GC_USA.

```
db2se create_srs <db_name> -srsName CALIFORNIA -srsId 101 -xScale 1
  -coordsysName NAD_1983_STATEPLANE_CALIFORNIA_I_FIPS_0401

SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE FUNCTION California_GC (
  street VARCHAR(128), city VARCHAR(128), zip VARCHAR(10))
 RETURNS db2gse.ST_Point
   LANGUAGE SQL
   RETURN db2gse.ST_Edge_GC_USA(street, city, 'CA', zip, 101,
     CAST(NULL AS INTEGER), CAST(NULL AS INTEGER),
     CAST(NULL AS DOUBLE), CAST(NULL AS VARCHAR(128)),
       CAST(NULL AS INTEGER), CAST(NULL AS VARCHAR(256)))

CREATE TABLE sample_geocoding (
  street VARCHAR(128),
  city  VARCHAR(128),
  state  VARCHAR(128),
    zip    VARCHAR(5) )

INSERT INTO geocoding(street, city, state, zip)
VALUES ('100 First North Street', 'San Jose', 'CA', NULL)

SELECT VARCHAR(ST_AsText(California_GC(street, city, zip)), 50)
FROM sample_geocoding
```

Resultados:

```
1
--------------------------------------------------
POINT ( 2004879.00000000 272723.00000000)

NetBIOS
```

Nota: Os valores das coordenadas X e Y do ponto são diferentes do primeiro exemplo porque é utilizado um sistema de referência espacial diferente.

ST_Endpoint

ST_Endpoint utiliza uma curva como um parâmetro de entrada e retorna o ponto que é o último ponto da curva. O ponto resultante é representado no sistema de referência espacial da curva especificada.

Se a curva especificada for nula ou vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

curve Um valor do tipo ST_Curve que representa a geometria a partir da qual o último ponto é retornado.

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

A instrução SELECT encontra o ponto extremo de cada uma das geometrias na tabela SAMPLE_LINES.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines(id INTEGER, line ST_Linestring)


INSERT INTO sample_lines VALUES
    (1, ST_LineString('linestring (10 10, 5 5, 0 0, 10 0, 5 5, 0 10)', 0) )

INSERT INTO sample_lines VALUES
    (2, ST_LineString('linestring z (0 0 4, 5 5 5, 10 10 6, 5 5 7)', 0) )


SELECT id, CAST(ST_AsText(ST_EndPoint(line)) as VARCHAR(50)) Endpoint
FROM   sample_lines
```

Resultados:

```
ID          ENDPOINT
----------- --------------------------------------------------
          1 POINT ( 0.00000000 10.00000000)
          2 POINT Z ( 5.00000000 5.00000000 7.00000000)
```

Referência Relacionada:

- "ST_PointN" na página 474

ST_Envelope

ST_Envelope utiliza uma geometria como um parâmetro de entrada e retorna um envelope em torno da geometria. O envelope é um retângulo que é representado como um polígono.

Se a geometria especificada for um ponto, uma cadeia de linhas horizontal ou uma cadeia de linhas vertical, será retornado um retângulo, que é ligeiramente maior do que a geometria especificada. Caso contrário, o retângulo limitador mínimo da geometria será retornado como envelope. Se a geometria especificada for nula ou vazia, então nulo é retornado. Para retornar o retângulo de limite mínimo exato para todas as geometrias, utilize a função ST_MBR.

Para dados geodésicos, o envelope é um polígono que inclui o círculo de limite mínimo da geometria.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry que representa a geometria para a qual o envelope será retornado.

Tipo de Retorno:

db2gse.ST_Polygon

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo cria várias geometrias e depois determina seus envelopes. Para o ponto não vazio e a cadeia de linhas (que é horizontal), o envelope é um retângulo que é ligeiramente maior do que a geometria.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
    (1, ST_Geometry('point EMPTY',0))

INSERT INTO sample_geoms VALUES
    (2, ST_Geometry('point zm (10 10 16 30)' ,0))

INSERT INTO sample_geoms VALUES
    (3, ST_Geometry('multipoint m (10 10 5, 50 10 6, 10 30 8)' ,0))

INSERT INTO sample_geoms VALUES
    (4, ST_Geometry('linestring (10 10, 20 10)',0))

INSERT INTO sample_geoms VALUES
    (5, ST_Geometry('polygon((40 120, 90 120, 90 150, 40 150, 40 120))',0))


SELECT id, CAST(ST_AsText(ST_Envelope(geometry)) as VARCHAR(160))  Envelope
FROM sample_geoms
```

Resultados:

```
ID          ENVELOPE
----------- ------------------------------------------------------------
          1 -

          2 POLYGON (( 9.00000000 9.00000000, 11.00000000 9.00000000,
   11.00000000 11.00000000, 9.00000000 11.00000000, 9.00000000 9.00000000))

          3 POLYGON (( 10.00000000 10.00000000, 50.00000000 10.00000000,
   50.00000000 30.00000000, 10.00000000 30.00000000, 10.00000000
   10.00000000))

          4 POLYGON (( 10.00000000 9.00000000, 20.00000000 9.00000000,
   20.00000000  11.00000000, 10.00000000 11.00000000, 10.00000000
   9.00000000))

          5 POLYGON (( 40.00000000 120.00000000, 90.00000000 120.00000000,
   90.00000000 150.00000000, 40.00000000 150.00000000, 40.00000000
   120.00000000))
```

Referência Relacionada:

- "ST_MBR" na página 433

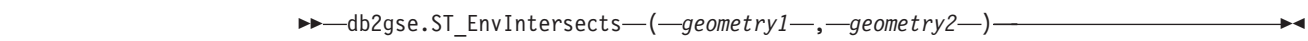
ST_EnvIntersects

ST_EnvIntersects utiliza duas geometrias como parâmetros de entrada e retorna 1 se os envelopes de duas geometrias forem interseccionados. Caso contrário, será retornado 0 (zero).

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Se uma das geometrias indicadas for nula ou vazia, o valor nulo será retornado.

Sintaxe:



Parâmetro:

geometry1
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria cujo envelope será testado quanto à interseção com o envelope de *geometry2*.

geometry2
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria cujo envelope será testado quanto à interseção com o envelope de *geometry1*.

Tipo de Retorno:

INTEGER

Exemplo:

Este exemplo cria duas cadeias de linha paralelas e verifica sua interseção. As próprias cadeias de linha não fazem interseção, mas seus envelopes sim.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('linestring (10 10, 50 50)',0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('linestring (10 20, 50 60)',0))

SELECT a.id, b.id, ST_Intersects(a.geometry, b.geometry) Intersects,
                   ST_EnvIntersects(a.geometry, b.geometry) Envelope_Intersects
FROM   sample_geoms a , sample_geoms b
WHERE  a.id = 1 and b.id=2
```

Resultados:

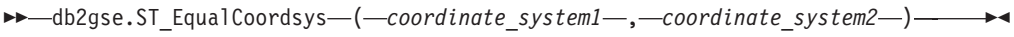
```
ID          ID          INTERSECTS  ENVELOPE_INTERSECTS
----------- ----------- ----------- -------------------
          1           2           0                   1
```

ST_EqualCoordsys

ST_EqualCoordsys utiliza duas definições do sistema de coordenadas como parâmetros de entrada e retorna o valor inteiro 1 (um) se as definições especificadas forem idênticas. Caso contrário, será retornado o valor inteiro 0 (zero). As definições do sistema de coordenadas são comparadas independentemente das diferenças de espaços, parênteses, caracteres maiúsculos e minúsculos e a representação de números de ponto flutuante.

Se qualquer uma das definições especificadas do sistema de coordenadas for nula, será retornado nulo.

Sintaxe:



Parâmetro:

coordinate_system1
Um valor do tipo VARCHAR(2048) que define o primeiro sistema de coordenadas a ser comparado com *coordinate_system2*.

coordinate_system2
Um valor do tipo VARCHAR(2048) que define o segundo sistema de coordenadas a ser comparado com *coordinate_system1*.

Tipo de Retorno:

INTEGER

Exemplo:

Este exemplo compara dois sistemas de coordenadas australianos para verificar se são iguais.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

VALUES ST_EqualCoordSys(
     (SELECT definition
      FROM   db2gse.ST_COORDINATE_SYSTEMS
      WHERE  coordsys_name='GCS_AUSTRALIAN') ,

     (SELECT definition
      FROM   db2gse.ST_COORDINATE_SYSTEMS
      WHERE  coordsys_name='GCS_AUSTRALIAN_1984')
      )
```

Resultados:

```
1
-----------
          0
```

Referência Relacionada:

• "A Exibição do Catálogo DB2GSE.ST_COORDINATE_SYSTEMS" na página 287

ST_Equals

ST_Equals utiliza duas geometrias como parâmetros de entrada e retorna 1 se as geometrias forem iguais. Caso contrário, será retornado 0 (zero). A ordem dos pontos utilizados para definir a geometria não é relevante para o teste de igualdade.

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Se qualquer uma das duas geometrias especificadas for nula, será retornado nulo.

Sintaxe:

Parâmetro:

geometry1
: Um valor do tipo ST_Geometry que representa a geometria que será comparada com *geometry2*.

geometry2
: Um valor do tipo ST_Geometry que representa a geometria que será comparada com *geometry1*.

Tipo de Retorno:

INTEGER

Exemplos:

Exemplo 1:

Este exemplo cria dois polígonos que têm suas coordenadas em uma ordem diferente. ST_Equal é utilizado para mostrar que estes polígonos são considerados iguais.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('polygon((50 30, 30 30, 30 50, 50 50, 50 30))' ,0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('polygon((50 30, 50 50, 30 50, 30 30, 50 30))' ,0))


SELECT a.id, b.id, ST_Equals(a.geometry, b.geometry) Equals
FROM sample_geoms a, sample_geoms b
WHERE a.id = 1 and b.id = 2
```

Resultados:

```
ID          ID          EQUALS
----------- ----------- -----------
          1           2           1
```

Exemplo 2:

Neste exemplo, duas geometrias são criadas com as mesmas coordenadas X e Y, mas com coordenadas M diferentes (medidas). Quando as geometrias são comparadas com a função ST_Equal, é retornado um 0 (zero) para indicar que estas geometrias não são iguais.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('multipoint m(80 80 6, 90 90 7)' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('multipoint m(80 80 6, 90 90 4)' ,0))


SELECT a.id, b.id, ST_Equals(a.geometry, b.geometry) Equals
FROM sample_geoms a, sample_geoms b
WHERE a.id = 3 and b.id = 4
```

Resultados:

```
ID          ID          EQUALS
----------- ----------- -----------
          3           4           0
```

Exemplo 3:

Neste exemplo, são criadas duas geometrias com um conjunto de coordenadas diferentes, mas ambas representam a mesma geometria. ST_Equal compara as geometrias e indica que ambas são realmente iguais.

```
SET current function path = current function path, db2gse
CREATE TABLE sample_geoms ( id INTEGER, geometry ST_Geometry )

INSERT INTO sample_geoms VALUES
   (5, ST_LineString('linestring ( 10 10, 40 40 )', 0)),
   (6, ST_LineString('linestring ( 10 10, 20 20, 40 40)', 0))
```

```
SELECT a.id, b.id, ST_Equals(a.geometry, b.geometry) Equals
FROM   sample_geoms a, sample_geoms b
WHERE  a.id = 5 AND b.id = 6
```

Resultados:

```
ID          ID          EQUALS
----------- ----------- -----------
          5           6           1
```

Referência Relacionada:

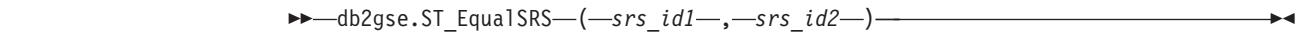
- "Funções que Comparam Recursos Geográficos" na página 308

ST_EqualSRS

ST_EqualSRS utiliza dois identificadores do sistema de referência espacial como parâmetros de entrada e retorna 1 se os sistemas de referência espacial especificados forem idênticos. Caso contrário, será retornado 0 (zero). Os deslocamentos, fatores de escala e sistemas de coordenadas são comparados.

Se qualquer um dos identificadores especificados do sistema de referência espacial for nulo, será retornado nulo.

Sintaxe:



Parâmetro:

srs_id1 Um valor do tipo INTEGER que identifica o primeiro sistema de referência espacial a ser comparado com o sistema de referência espacial identificado por *srs_id2*.

srs_id2 Um valor do tipo INTEGER que identifica o segundo sistema de referência espacial a ser comparado com o sistema de referência espacial identificado por *srs_id1* .

Tipo de Retorno:

INTEGER

Exemplo:

São criados dois sistemas de referência espacial similares com as seguintes chamadas para db2se.

```
db2se create_srs SAMP_DB  -srsId 12 -srsName NYE_12 -xOffset 0 -yOffset 0
   -xScale 1 -yScale 1 -coordsysName
   NAD_1983_StatePlane_New_York_East_FIPS_3101_Feet

db2se create_srs SAMP_DB  -srsId 22 -srsName NYE_22 -xOffset 0 -yOffset 0
   -xScale 1 -yScale 1 -coordsysName
   NAD_1983_StatePlane_New_York_East_FIPS_3101_Feet
```

Estes SRSs têm os mesmos valores de deslocamento e escala e se referem aos mesmos sistemas de coordenadas. A única diferença está no nome definido e no ID do SRS. Portanto, a comparação retorna 1, que indica que eles são iguais.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

VALUES ST_EqualSRS(12, 22)
```

Resultados:

```
1
-----------
          1
```

Referência Relacionada:

- "A Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS" na página 295

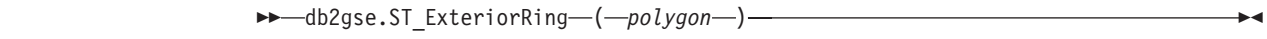
ST_ExteriorRing

ST_ExteriorRing utiliza um polígono como um parâmetro de entrada e retorna seu anel externo como uma curva. A curva resultante é representada como o sistema de referência espacial do polígono especificado.

Se o polígono especificado for nulo ou vazio, será retornado nulo. Se o polígono não tiver anéis internos, o anel externo retornado será idêntico ao limite do polígono.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

polygon
Um valor do tipo ST_Polygon que representa o polígono para o qual o anel externo será retornado.

Tipo de Retorno:

db2gse.ST_Curve

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo cria dois polígonos, um com dois anéis internos e outro sem anéis internos, em seguida, determina seus anéis externos.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys VALUES
       (1, ST_Polygon('polygon((40 120, 90 120, 90 150, 40 150, 40 120),
                     (50 130, 60 130, 60 140, 50 140, 50 130),
                     (70 130, 80 130, 80 140, 70 140, 70 130))' ,0))

INSERT INTO sample_polys VALUES
```

```
       (2, ST_Polygon('polygon((10 10, 50 10, 10 30, 10 10))' ,0))


SELECT id, CAST(ST_AsText(ST_ExteriorRing(geometry))
   AS VARCHAR(180)) Exterior_Ring
FROM sample_polys
```

Resultados:

```
ID          EXTERIOR_RING
----------- ------------------------------------------------------------------
          1 LINESTRING ( 40.00000000 120.00000000, 90.00000000
   120.00000000,  90.00000000 150.00000000, 40.00000000 150.00000000,
   40.00000000 120.00000000)

          2 LINESTRING ( 10.00000000 10.00000000, 50.00000000
   10.00000000, 10.00000000 30.00000000, 10.00000000 10.00000000)
```

Referência Relacionada:

- "ST_Boundary" na página 356

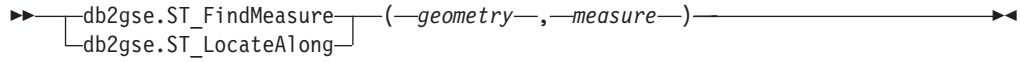
ST_FindMeasure ou ST_LocateAlong

ST_FindMeasure ou ST_LocateAlong utiliza uma geometria e uma medida como parâmetros de entrada e retorna um multiponto ou multicurva dessa parte da geometria especificada que tem exatamente a medida especificada da geometria especificada que contém a medida especificada. Para pontos e multipontos, todos os pontos com a medida especificada são retornados. Para curvas, multicurvas, superfícies e multisuperfícies, a interpolação é executada para calcular o resultado. O cálculo para superfícies e multisuperfícies é executado no limite da geometria.

Para pontos e multipontos, se a medida especificada não for encontrada, será retornada uma geometria vazia. Para as demais geometrias, se a medida especificada for menor do que a menor medida na geometria ou maior do que a maior medida da geometria, será retornada uma geometria vazia. Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria na qual serão procuradas partes cujas coordenadas M (medidas) contêm *measure*.

measure
: Um valor do tipo DOUBLE que é a medida cujas partes da *geometria* devem ser incluídas no resultado.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplos:

A seguinte instrução CREATE TABLE cria a tabela SAMPLE_GEOMETRIES. SAMPLE_GEOMETRIES tem duas colunas: a coluna do ID, que identifica exclusivamente cada linha e a coluna GEOMETRY ST_Geometry, que armazena a geometria de exemplo.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries(id SMALLINT, geometry ST_GEOMETRY)
```

A seguinte instrução INSERT insere duas linhas. A primeira é uma cadeia de linhas; a segunda é um multiponto.

```
INSERT INTO sample_geometries(id, geometry)
VALUES
  (1, ST_LineString('linestring m (2 2 3, 3 5 3, 3 3 6, 4 4 8)', 1)),
  (2, ST_MultiPoint('multipoint m
      (2 2 3, 3 5 3, 3 3 6, 4 4 6, 5 5 6, 6 6 8)', 1))
```

Exemplo 1:

Na seguinte instrução SELECT e no conjunto de resultados correspondente, a função ST_FindMeasure é direcionada para localizar pontos cuja medida seja 7. A primeira linha retorna um ponto. No entanto, a segunda linha retorna um ponto vazio. Para recursos lineares (geometria com uma dimensão maior do que 0), ST_FindMeasure poderá interpolar o ponto; porém, para multipontos, a medida de destino deve ter uma correspondência exata.

```
SELECT id, cast(ST_AsText(ST_FindMeasure(geometry, 7))
   AS varchar(45)) AS measure_7
FROM   sample_geometries
```

Resultados:

```
ID     MEASURE_7
------ ---------------------------------------------
     1 POINT M ( 3.50000000 3.50000000 7.00000000)
     2 POINT EMPTY
```

Exemplo 2:

Na seguinte instrução SELECT e no conjunto de resultados correspondente, a função ST_FindMeasure retorna um ponto e um multiponto. A medida de destino 6 corresponde às medidas nos dados de origem de ST_FindMeasure e no multiponto.

```
SELECT id, cast(ST_AsText(ST_FindMeasure(geometry, 6))
   AS varchar(120)) AS measure_6
FROM   sample_geometries
```

Resultados:

```
ID     MEASURE_6
------ ------------------------------------------------------------
1      POINT M ( 3.00000000 3.00000000 6.00000000)

2      MULTIPOINT M ( 3.00000000 3.00000000 6.00000000, 4.00000000
        4.00000000 6.00000000, 5.00000000 5.00000000 6.00000000)
```

Referência Relacionada:

- "ST_MeasureBetween, ST_LocateBetween" na página 435

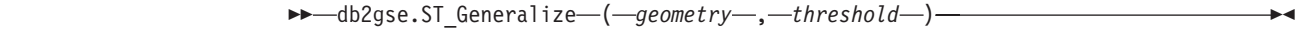
ST_Generalize

ST_Generalize utiliza uma geometria e um limite como parâmetros de entrada e representa a geometria especificada com um número reduzido de pontos, ao mesmo tempo que preserva as características gerais da geometria. O algoritmo de simplificação de linha Douglas-Peucker é utilizado, através do qual a seqüência de pontos que definem a geometria é recursivamente subdivida até que uma sucessão dos pontos possa ser substituída por um segmento linear reto. Neste segmento de linha, nenhum dos pontos de definição é desviado do segmento de linha reto além do limite determinado. As coordenadas Z e M não são consideradas para a simplificação. A geometria resultante está no sistema de referência espacial da geometria especificada.

Se a geometria especificada for vazia, uma geometria vazia de tipo ST_Point será retornada. Se a geometria especificada ou o limite for nulo, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria à qual a simplificação de linha é aplicada.

threshold
Um valor do tipo DOUBLE que identifica o limite a ser utilizado para o algoritmo de simplificação de linha. O limite deve ser maior ou igual a 0 (zero). Quanto maior o limite, menor o número de pontos que serão utilizados para representar a geometria generalizada. Para dados geodésicos, a unidade para o limite está em metros.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplos:

Nos exemplos a seguir, os resultados foram reformatados para possibilitar a leitura. O espaçamento nos resultados irá variar de acordo com a exibição.

Uma cadeia de linhas é criada com oito pontos que vão de (10, 10) a (80, 80). O caminho é quase uma linha reta, mas alguns dos pontos estão um pouco fora da linha. A função ST_Generalize pode ser utilizada para reduzir o número de pontos na linha.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, geometry ST_LineString)

INSERT INTO sample_lines VALUES
       (1, ST_Linestring('linestring(10 10, 21 20, 34 26, 40 40,
                                     52 50, 59 63, 70 71, 80 80)' ,0))
```

Exemplo 1:

Quando um fator de generalização 3 é utilizado, a cadeia de linhas é reduzida para quatro coordenadas e ainda fica muito próxima da representação original da cadeia de linhas.

```
SELECT CAST(ST_AsText(ST_Generalize(geometry, 3)) as VARCHAR(115))
  Generalize_3
FROM sample_lines
```

Resultados:

```
GENERALIZE 3
------------------------------------------------------------------------
LINESTRING ( 10.00000000 10.00000000, 34.00000000 26.00000000,
  59.00000000  63.00000000, 80.00000000 80.00000000)
```

Exemplo 2:

Quando é utilizado um fator de generalização 6, a cadeia de linhas é reduzida para somente duas coordenadas. Isto gera uma cadeia de linhas mais simples do que o exemplo anterior, porém, se desvia mais da representação original.

```
SELECT CAST(ST_AsText(ST_Generalize(geometry, 6)) as VARCHAR(65))
  Generalize_6
FROM sample_lines
```

Resultados:

```
GENERALIZE 6
-----------------------------------------------------------------
LINESTRING ( 10.00000000 10.00000000, 80.00000000 80.00000000)
```

ST_GeomCollection

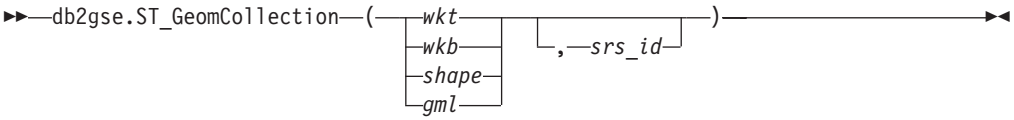
ST_GeomCollection constrói uma coleção de geometrias a partir de uma das seguintes entradas:

- Uma representação de texto reconhecida
- Uma representação binária reconhecida
- Uma representação de formatos ESRI
- Uma representação na GML (Geography Markup Language)

Um identificador do sistema de referência espacial opcional pode ser especificado para identificar o sistema de referência espacial no qual a coleção de geometrias resultante está localizado.

Se a representação de texto reconhecida, a representação binária reconhecida, a representação de formatos ESRI ou a representação GML for nula, será retornado nulo.

Sintaxe:



Parâmetro:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida da coleção de geometrias resultantes.

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida da coleção de geometrias resultante.

shape Um valor do tipo BLOB(2G) que representa a representação de formatos ESRI da coleção de geometrias resultantes.

gml Um valor do tipo CLOB(2G) que representa a coleção de geometrias resultantes utiizando a GML (Geography Markup Language).

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da coleta de geometria resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado implicitamente.

Se *srs_id* não identifica um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então um erro é retornado (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_GeomCollection

Observações:

Se o parâmetro *srs_id* for omitido, poderá ser necessário converter *wkt* e *gml* explicitamente no tipo de dados CLOB. Caso contrário, o DB2 pode ser resolvido para a função utilizada para converter valores do tipo de referência REF(ST_GeomCollection) no tipo ST_GeomCollection. O exemplo a seguir assegura que o DB2 será resolvido para a função correta:

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir ilustra como a função ST_GeomCollection pode ser utilizada para criar e inserir um multiponto, multilinha e multipolígono a partir de representação WKT (Well-Known Text) e um multiponto a partir da GML (Geographic Markup Language) em uma coluna GeomCollection.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geomcollections(id INTEGER,
   geometry ST_GEOMCOLLECTION)

INSERT INTO sample_geomcollections(id, geometry)
VALUES
    (4001, ST_GeomCollection('multipoint(1 2, 4 3, 5 6)', 1) ),
    (4002, ST_GeomCollection('multilinestring(
                              (33 2, 34 3, 35 6),
                              (28 4, 29 5, 31 8, 43 12),
                              (39 3, 37 4, 36 7))', 1) ),
    (4003, ST_GeomCollection('multipolygon(((3 3, 4 6, 5 3, 3 3),
                              (8 24, 9 25, 1 28, 8 24),
                              (13 33, 7 36, 1 40, 10 43, 13 33)))', 1)),
    (4004, ST_GeomCollection('<gml:MultiPoint srsName="EPSG:4269"
                              ><gml:PointMember><gml:Point>
                              <gml:coord><gml:X>10</gml:X>
```

```
                                <gml:Y>20</gml:Y></gml: coord></gml:Point>
                                </gml:PointMember><gml:PointMember>
                                <gml:Point><gml:coord><gml:X>30</gml:X>
                                <gml:Y>40</gml:Y></gml:coord></gml:Point>
                                </gml:PointMember></gml:MultiPoint>', 1))

SELECT id, cast(geometry..ST_AsText AS varchar(350)) AS geomcollection
FROM   sample_geomcollections
```

Resultados:

```
ID          GEOMCOLLECTION
----------- ------------------------------------------------------------
4001         MULTIPOINT ( 1.00000000 2.00000000, 4.00000000 3.00000000,
    5.00000000 6.00000000)

4002         MULTILINESTRING (( 33.00000000 2.00000000, 34.00000000
   3.00000000, 35.00000000 6.00000000),( 28.00000000 4.00000000,
   29.00000000 5.00000000, 31.00000000 8.00000000, 43.00000000
   12.00000000),(39.00000000 3.00000000, 37.00000000 4.00000000,
   36.00000000 7.00000000))

4003         MULTIPOLYGON ((( 13.00000000 33.00000000, 10.00000000
    43.00000000, 1.00000000 40.00000000, 7.00000000 36.00000000,
    13.00000000 33.00000000)),(( 8.00000000 24.00000000, 9.00000000
    25.00000000, 1.00000000 28.00000000, 8.00000000 24.00000000)),
    (( 3.00000000 3.00000000,5.00000000 3.00000000, 4.00000000
    6.00000000,3.00000000 3.00000000)))

4004         MULTIPOINT ( 10.00000000 20.00000000, 30.00000000
    40.00000000)
```

Referência Relacionada:

- "Representação WKT (Well-Known Text)" na página 519
- "Representação WKB (Well-Known Binary)" na página 524
- "Representação de Formatos" na página 526
- "Representação GML (Geography Markup Language)" na página 526

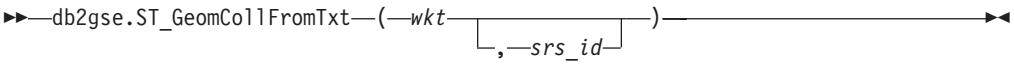
ST_GeomCollFromTxt

ST_GeomCollFromTxt utiliza uma representação de texto reconhecida de uma coleção de geometrias e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna a coleção de geometrias correspondentes.

Se a representação de texto reconhecida for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é ST_GeomCollection. Ela é recomendada por sua flexibilidade: ST_GeomCollection utiliza formatos adicionais de entrada, além da representação binária reconhecida.

Sintaxe:



Parâmetro:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida da coleção de geometrias resultantes.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da coleta de geometria resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado implicitamente.

Se *srs_id* não identifica um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então um erro é retornado (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_GeomCollection

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir ilustra como a função ST_GeomCollFromTxt pode ser utilizada para criar e inserir um multiponto, multilinha e um multipolígono a partir de uma representação de texto reconhecida (WKT) em uma coluna GeomCollection.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geomcollections(id INTEGER, geometry ST_GEOMCOLLECTION)

INSERT INTO sample_geomcollections(id, geometry)
VALUES
    (4011, ST_GeomCollFromTxt('multipoint(1 2, 4 3, 5 6)', 1) ),
    (4012, ST_GeomCollFromTxt('multilinestring(
                           (33 2, 34 3, 35 6),
                           (28 4, 29 5, 31 8, 43 12),
                           (39 3, 37 4, 36 7))', 1) ),
    (4013, ST_GeomCollFromTxt('multipolygon(((3 3, 4 6, 5 3, 3 3),
                           (8 24, 9 25, 1 28, 8 24),
                           (13 33, 7 36, 1 40, 10 43, 13 33)))', 1))

SELECT id, cast(geometry..ST_AsText AS varchar(340))
   AS geomcollection
FROM   sample_geomcollections
```

Resultados:

```
ID          GEOMCOLLECTION
----------- ------------------------------------------------------------------
4011        MULTIPOINT ( 1.00000000 2.00000000, 4.00000000 3.00000000,
   5.00000000 6.00000000)

4012        MULTILINESTRING (( 33.00000000 2.00000000, 34.00000000
   3.00000000, 35.00000000 6.00000000),( 28.00000000 4.00000000, 29.00000000
   5.00000000, 31.00000000 8.00000000, 43.00000000 12.00000000),( 39.00000000
   3.00000000, 37.00000000 4.00000000, 36.00000000 7.00000000))

4013        MULTIPOLYGON ((( 13.00000000 33.00000000, 10.00000000 43.00000000,
   1.00000000 40.00000000, 7.00000000 36.00000000, 13.00000000 33.00000000)),
   (( 8.00000000 24.00000000, 9.00000000 25.00000000, 1.00000000 28.00000000,
   8.00000000 24.00000000)),(( 3.00000000 3.00000000, 5.00000000 3.00000000,
   4.00000000 6.00000000, 3.00000000 3.00000000)))
```

Referência Relacionada:

• "ST_GeomCollection" na página 392

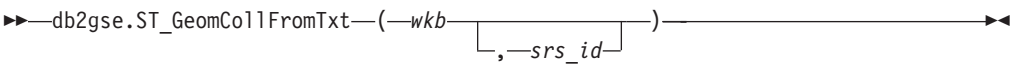
ST_GeomCollFromWKB

ST_GeomCollFromWKB utiliza uma representação binária reconhecida de uma coleção de geometrias e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna a coleção de geometrias correspondentes.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

A versão preferida para esta funcionalidade é ST_GeomCollection.

Sintaxe:



Parâmetro:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida da coleção de geometrias resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da coleta de geometria resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado implicitamente.

Se *srs_id* não identifica um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então um erro é retornado (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_GeomCollection

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir ilustra como a função ST_GeomCollFromWKB pode ser utilizada para criar e consultar as coordenadas de uma coleção de geometrias em uma representação binária reconhecida. As linhas são inseridas na tabela SAMPLE_GEOMCOLLECTION com IDs 4021 e 4022 e as coleções de geometrias no sistema de referência espacial 1.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geomcollections(id INTEGER,
  geometry ST_GEOMCOLLECTION, wkb BLOB(32k))

INSERT INTO sample_geomcollections(id, geometry)
VALUES
  (4021, ST_GeomCollFromTxt('multipoint(1 2, 4 3, 5 6)', 1)),
  (4022, ST_GeomCollFromTxt('multilinestring(
                            (33 2, 34 3, 35 6),
                            (28 4, 29 5, 31 8, 43 12))', 1))

UPDATE sample_geomcollections AS temp_correlated
```

```
SET    wkb = geometry..ST_AsBinary
WHERE  id = temp_correlated.id

SELECT id, cast(ST_GeomCollFromWKB(wkb)..ST_AsText
   AS varchar(190)) AS GeomCollection
FROM   sample_geomcollections
```

Resultados:

```
ID          GEOMCOLLECTION
----------- ----------------------------------------------------
       4021 MULTIPOINT ( 1.00000000 2.00000000, 4.00000000
  3.00000000, 5.00000000 6.00000000)

       4022 MULTILINESTRING (( 33.00000000 2.00000000,
 34.00000000 3.00000000, 35.00000000 6.00000000),( 28.00000000
 4.00000000, 29.00000000 5.00000000, 31.00000000 8.00000000,
 43.00000000 12.00000000))
```

Referência Relacionada:

- "Representação WKB (Well-Known Binary)" na página 524

ST_Geometry

ST_Geometry constrói uma geometria a partir de uma das seguintes entradas:

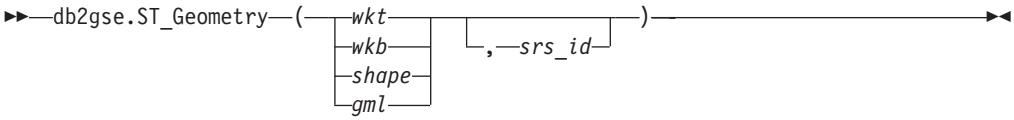
- Uma representação de texto reconhecida
- Uma representação binária reconhecida
- Uma representação de formatos ESRI
- Uma representação na GML (Geography Markup Language)

Um identificador do sistema de referência espacial opcional pode ser especificado para identificar o sistema de referência espacial no qual a geometria resultante está localizada.

O tipo dinâmico da geometria resultante é um dos subtipos instanciáveis de ST_Geometry.

Se a representação de texto reconhecida, a representação binária reconhecida, a representação de formatos ESRI ou a representação GML for nula, será retornado nulo.

Sintaxe:



Parâmetro:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida da geometria resultante.

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida da geometria resultante.

shape Um valor do tipo BLOB(2G) que representa a representação de formatos ESRI da geometria resultante.

gml Um valor do tipo CLOB(2G) que representa a geometria resultante utilizando a GML (Geography Markup Language).

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da geometria resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado implicitamente.

Se *srs_id* não identifica um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então um erro é retornado (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir ilustra como a função ST_Geometry pode ser utilizada para criar e inserir um ponto a partir de uma representação de ponto de WKT (Well-Known Text) ou linha de uma representação de linha de GML (Geographic Markup Language).

A função ST_Geometry é a mais flexível das funções do construtor de tipos espaciais porque ela pode criar qualquer tipo espacial a partir de várias representações geométricas. ST_LineFromText pode criar apenas uma linha a partir da representação de linha WKT. ST_WKTToSql pode construir qualquer tipo, mas somente a partir da representação WKT.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries(id INTEGER, geometry ST_GEOMETRY)

INSERT INTO sample_geometries(id, geometry)
VALUES
    (7001, ST_Geometry('point(1 2)', 1) ),
    (7002, ST_Geometry('linestring(33 2, 34 3, 35 6)', 1) ),
    (7003, ST_Geometry('polygon((3 3, 4 6, 5 3, 3 3))', 1)),
    (7004, ST_Geometry('<gml:Point srsName=";EPSG:4269";><gml:coord>
         <gml:X>50</gml:X><gml:Y>60</gml:Y></gml:coord>
           </gml:Point>', 1))

SELECT id, cast(geometry..ST_AsText AS varchar(120)) AS geometry
FROM   sample_geometries
```

Resultados:

```
ID          GEOMETRY
----------- ------------------------------------------------------------
7001        POINT ( 1.00000000 2.00000000)

7002        LINESTRING ( 33.00000000 2.00000000, 34.00000000 3.00000000,
  35.00000000 6.00000000)

7003        POLYGON (( 3.00000000 3.00000000, 5.00000000 3.00000000,
  4.00000000 6.00000000, 3.00000000 3.00000000))

7004        POINT ( 50.00000000 60.00000000)
```

Referência Relacionada:

- "Representação WKT (Well-Known Text)" na página 519

ST_GeometryN

ST_GeometryN utiliza uma coleção de geometria e um índice como parâmetros de entrada e retorna a geometria na coleção que é identificada pelo índice. A geometria resultante é representada no sistema de referência espacial da coleção de geometria especificada.

Se a coleção de geometrias especificada for nula ou vazia ou se o índice for menor do que 1 ou maior do que o número de geometrias na coleção, será retornado nulo e ocorrerá uma condição de aviso (01HS0).

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

collection
: Um valor do tipo ST_GeomCollection ou um de seus subtipos que representa a coleção de geometrias para localizar a *ú*ltima geometria.

index
: Um valor do tipo INTEGER que identifica a *ú*ltima geometria que deve ser retornada da *collection*.

  Se *index* for menor do que 1 ou maior do que o número de geometrias na coleção, será retornado nulo e também um aviso (SQLSTATE 01HS0).

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

O código a seguir ilustra como escolher a segunda geometria em uma coleção de geometrias.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geomcollections (id INTEGER,
  geometry ST_GEOMCOLLECTION)

INSERT INTO sample_geomcollections(id, geometry)
VALUES
    (4001, ST_GeomCollection('multipoint(1 2, 4 3)', 1) ),
    (4002, ST_GeomCollection('multilinestring(
                            (33 2, 34 3, 35 6),
                            (28 4, 29 5, 31 8, 43 12),
                            (39 3, 37 4, 36 7))', 1) ),
    (4003, ST_GeomCollection('multipolygon(((3 3, 4 6, 5 3, 3 3),
                               (8 24, 9 25, 1 28, 8 24),
                               (13 33, 7 36, 1 40, 10 43, 13 33)))', 1))

SELECT id, cast(ST_GeometryN(geometry, 2)..ST_AsText AS varchar(110))
    AS second_geometry
FROM   sample_geomcollections
```

Resultados:

```
ID          SECOND_GEOMETRY
----------- ------------------------------------------------------------
       4001 POINT ( 4.00000000 3.00000000)

       4002 LINESTRING ( 28.00000000 4.00000000, 29.00000000 5.00000000,
   31.00000000 8.00000000, 43.00000000 12.00000000)

       4003 POLYGON (( 8.00000000 24.00000000, 9.00000000 25.00000000,
   1.00000000 28.00000000, 8.00000000 24.00000000))
```

Referência Relacionada:

- "ST_NumGeometries" na página 458

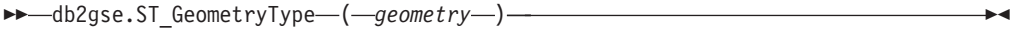
ST_GeometryType

ST_GeometryType utiliza uma geometria como parâmetro de entrada e retorna o nome completo do tipo dinâmico dessa geometria.

As funções TYPE_SCHEMA e TYPE_NAME do DB2 têm o mesmo efeito.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry para o qual o tipo de geometria será retornado.

Tipo de Retorno:

VARCHAR(128)

Exemplos:

O código a seguir ilustra como determinar o tipo de uma geometria.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries (id INTEGER, geometry ST_GEOMETRY)

INSERT INTO sample_geometries(id, geometry)
VALUES
    (7101, ST_Geometry('point(1 2)', 1) ),
    (7102, ST_Geometry('linestring(33 2, 34 3, 35 6)', 1) ),
    (7103, ST_Geometry('polygon((3 3, 4 6, 5 3, 3 3))', 1)),
    (7104, ST_Geometry('multipoint(1 2, 4 3)', 1) )

SELECT id, geometry..ST_GeometryType AS geometry_type
FROM   sample_geometries
```

Resultados:

```
ID          GEOMETRY_TYPE
----------- ------------------------------
       7101 "DB2GSE  "."ST_POINT"
       7102 "DB2GSE  "."ST_LINESTRING"
       7103 "DB2GSE  "."ST_POLYGON"
       7104 "DB2GSE  "."ST_MULTIPOINT"
```

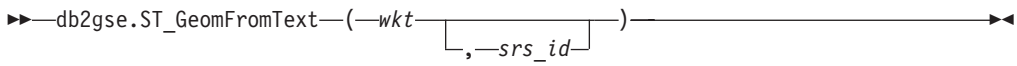
ST_GeomFromText

ST_GeomFromText utiliza uma representação de texto reconhecida de uma geometria e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna a geometria correspondente.

Se a representação de texto reconhecida for nula, então nulo é retornado.

A versão preferida para esta funcionalidade é ST_Geometry.

Sintaxe:



Parâmetro:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida da geometria resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da geometria resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado implicitamente.

Se *srs_id* não identifica um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então um erro é retornado (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Neste exemplo, a função ST_GeomFromText é utilizada para criar e inserir um ponto a partir de uma representação de ponto de WKT (Well-Known Text).

O código a seguir insere linhas na tabela SAMPLE_POINTS com IDs e geometrias no sistema de referência espacial 1 utilizando a representação WKT.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries(id INTEGER, geometry ST_GEOMETRY)

INSERT INTO sample_geometries(id, geometry)
VALUES
```

```
    (1251, ST_GeomFromText('point(1 2)', 1) ),
    (1252, ST_GeomFromText('linestring(33 2, 34 3, 35 6)', 1) ),
    (1253, ST_GeomFromText('polygon((3 3, 4 6, 5 3, 3 3))', 1))
```

A instrução SELECT a seguir retornará o ID e GEOMETRIES a partir da tabela SAMPLE_GEOMETRIES.

```
SELECT id, cast(geometry..ST_AsText AS varchar(105))
   AS geometry
FROM   sample_geometries
```

Resultados:

```
ID          GEOMETRY
----------- ------------------------------------------------------------
       1251 POINT ( 1.00000000 2.00000000)

       1252 LINESTRING ( 33.00000000 2.00000000, 34.00000000 3.00000000,
              35.00000000 6.00000000)

       1253 POLYGON (( 3.00000000 3.00000000, 5.00000000 3.00000000,
              4.00000000 6.00000000, 3.00000000 3.00000000))
```

Referência Relacionada:

- "Representação WKT (Well-Known Text)" na página 519

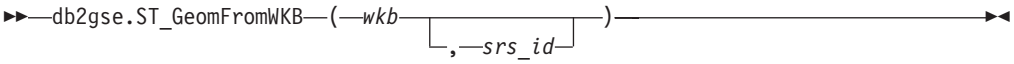
ST_GeomFromWKB

ST_GeomFromWKB utiliza uma representação binária reconhecida de uma geometria e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna a geometria correspondente.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

A versão preferida para esta funcionalidade é ST_Geometry.

Sintaxe:



Parâmetro:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida da geometria resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da geometria resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado implicitamente.

Se o parâmetro srs_id especificado não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, será retornado um erro (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Geometry

Exemplos:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir ilustra como a função ST_GeomFromWKB pode ser utilizada para criar e inserir uma linha a partir de uma representação de linha binária reconhecida (WKB).

O exemplo a seguir insere um registro na tabela SAMPLE_GEOMETRIES com um ID e uma geometria no sistema de referência espacial 1 em uma representação WKB.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries (id INTEGER, geometry ST_GEOMETRY,
    wkb BLOB(32K))

INSERT INTO sample_geometries(id, geometry)
VALUES
    (1901, ST_GeomFromText('point(1 2)', 1) ),
    (1902, ST_GeomFromText('linestring(33 2, 34 3, 35 6)', 1) ),
    (1903, ST_GeomFromText('polygon((3 3, 4 6, 5 3, 3 3))', 1))

UPDATE sample_geometries AS temp_correlated
SET    wkb = geometry..ST_AsBinary
WHERE  id = temp_correlated.id

SELECT id, cast(ST_GeomFromWKB(wkb)..ST_AsText AS varchar(190))
   AS geometry
FROM   sample_geometries
```

Resultados:

```
 ID          GEOMETRY
 ----------- ------------------------------------------------------------
1901         POINT ( 1.00000000 2.00000000)

1902         LINESTRING ( 33.00000000 2.00000000, 34.00000000
   3.00000000, 35.00000000 6.00000000)

1903         POLYGON (( 3.00000000 3.00000000, 5.00000000 3.00000000,
   4.00000000 6.00000000, 3.00000000 3.00000000))
```

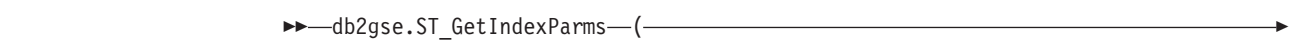
Referência Relacionada:

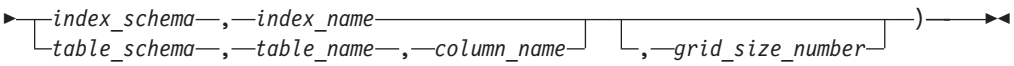
- "Representação WKB (Well-Known Binary)" na página 524

ST_GetIndexParms

ST_GetIndexParms utiliza o identificador para um índice espacial ou para uma coluna espacial como parâmetro de entrada e retorna os parâmetros utilizados para definir o índice ou o índice na coluna espacial. Se for especificado um número de parâmetro adicional, somente o tamanho de grade identificado pelo número será retornado.

Sintaxe:





Parâmetro:

index_schema
Um valor do tipo VARCHAR(128) que identifica o esquema no qual o índice espacial com o nome não qualificado *index_name* está. O nome do esquema faz distinção entre maiúsculas e minúsculas e deve ser listado na exibição do catálogo SYSCAT.SCHEMATA.

Se este parâmetro for nulo, o valor do registro especial CURRENT SCHEMA será utilizado como o nome do esquema para o índice espacial.

index_name
Um valor do tipo VARCHAR(128) que contém o nome não qualificado do índice espacial para o qual os parâmetros do índice são retornados. O nome do índice faz distinção entre maiúsculas e minúsculas e deve ser listado na exibição do catálogo SYSCAT.INDEXES para o esquema *index_schema*.

table_schema
Um valor do tipo VARCHAR(128) que identifica o esquema no qual a tabela com o nome não qualificado *table_name* está. O nome do esquema faz distinção entre maiúsculas e minúsculas e deve ser listado na exibição do catálogo SYSCAT.SCHEMATA.

Se este parâmetro for nulo, o valor do registro especial CURRENT SCHEMA será utilizado como o nome do esquema para o índice espacial.

table_name
Um valor do tipo VARCHAR(128) que contém o nome não qualificado da tabela com a coluna espacial *column_name*. O nome da tabela faz distinção entre maiúsculas e minúsculas e deve ser listado na exibição do catálogo SYSCAT.TABLES para o esquema *table_schema*.

column_name
Um valor do tipo VARCHAR(128) que identifica a coluna na tabela *table_schema.table_name* para a qual os parâmetros de índice do índice espacial dessa coluna são retornados. O nome da coluna faz distinção entre maiúsculas e minúsculas e deve ser listado na exibição do catálogo SYSCAT.COLUMNS para a tabela *table_schema.table_name*.

Se não houver nenhum índice espacial definido na coluna, ocorrerá um erro (SQLSTATE 38SQ0).

grid_size_number
Um valor DOUBLE que identifica o parâmetro cujo valor ou valores devem ser retornados.

Se este valor for menor do que 1 ou maior do que 3, ocorrerá um erro (SQLSTATE 38SQ1).

Tipo de Retorno:

DOUBLE (se *grid_size_number* for especificado)

Se *grid_size_number* não for especificado, será retornada uma tabela com as duas colunas ORDINAL e VALUE. A coluna ORDINAL será do tipo INTEGER e a coluna VALUE será do tipo DOUBLE.

Se os parâmetros forem retornados para um índice de grade, a coluna ORDINAL conterá os valores 1, 2 e 3 para os tamanhos da primeira, segunda e terceira grades, respectivamente. A coluna VALUE contém os tamanhos de grades.

A coluna VALUE contém os respectivos valores para cada um dos parâmetros.

Exemplos:

Este código cria uma tabela com uma coluna espacial e um índice espacial.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sch.offices (name VARCHAR(30), location ST_Point )

 CREATE INDEX sch.idx ON sch.offices(location)
      EXTEND USING db2gse.spatial_index(1e0, 10e0, 1000e0)
```

A função ST_GetIndexParms pode ser utilizada para recuperar os valores para os parâmetros que foram utilizados quando o índice espacial foi criado.

Exemplo 1:

Este exemplo mostra como recuperar os três tamanhos de grades para um índice de grade espacial separadamente, especificando de forma explícita qual parâmetro, identificado por seu número, deve ser retornado.

```
VALUES ST_GetIndexParms('SCH', 'ESCRITÓRIOS', 'LOCALIZAÇÃO', 1)
```

Resultados:

```
1
------------------------
  +1.00000000000000E+000

VALUES ST_GetIndexParms('SCH', 'ESCRITÓRIOS', 'LOCALIZAÇÃO', 2)
```

Resultados:

```
1
------------------------
  +1.00000000000000E+001
VALUES ST_GetIndexParms('SCH', 'IDX', 3)
```

Resultados:

```
1
------------------------
  +1.00000000000000E+003
```

Exemplo 2:

Este exemplo mostra como recuperar todos os parâmetros de um índice de grade espacial. A função ST_GetIndexParms retorna uma tabela que indica o número do parâmetro e o tamanho da grade correspondente.

```
SELECT * FROM TABLE ( ST_GetIndexParms('SCH', 'ESCRITÓRIOS', 'LOCALIZAÇÃO') ) AS t
```

Resultados:

```
ORDINAL     VALUE
----------- ------------------------
          1   +1.00000000000000E+000
          2   +1.00000000000000E+001
          3   +1.00000000000000E+003
SELECT * FROM TABLE ( ST_GetIndexParms('SCH', 'IDX') ) AS t
```

Resultados:

```
ORDINAL     VALUE
----------- ----------------------
          1   +1.00000000000000E+000
          2   +1.00000000000000E+001
          3   +1.00000000000000E+003
```

Conceitos Relacionados:

- "Índices de Grade Espaciais" na página 102

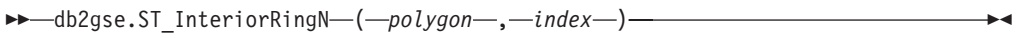
ST_InteriorRingN

ST_InteriorRingN utiliza um polígono e um índice como parâmetros de entrada e retorna o anel interno identificado pelo índice especificado como uma cadeia de linhas. Os anéis internos são organizados de acordo com as regras definidas pelas rotinas de verificação de geometria interna.

Se o polígono fornecido for nulo ou vazio, ou se não tiver nenhum anel interno, será retornado nulo. Se o índice for menor do que 1 ou maior do que o número de anéis internos no polígono, será retornado nulo e ocorrerá uma condição de aviso (1HS1).

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

polygon
: Um valor do tipo ST_Polygon que representa a geometria a partir da qual o anel interno identificado por *index* é retornado.

index
: Um valor do tipo INTEGER que identifica o *ú*ltimo anel interno retornado. Se não houver nenhum anel interno identificado por *index*, ocorrerá uma condição de aviso (01HS1).

Tipo de Retorno:

db2gse.ST_Curve

Exemplo:

Neste exemplo, é criado um polígono com dois anéis internos. A chamada ST_InteriorRingN é então utilizada para recuperar o segundo anel interno.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys VALUES
       (1, ST_Polygon('polygon((40 120, 90 120, 90 150, 40 150, 40 120),
                      (50 130, 60 130, 60 140, 50 140, 50 130),
                      (70 130, 80 130, 80 140, 70 140, 70 130))' ,0))


SELECT id, CAST(ST_AsText(ST_InteriorRingN(geometry, 2)) as VARCHAR(180))
       Interior_Ring
FROM sample_polys
```

Resultados:

```
ID          INTERIOR_RING
----------- ----------------------------------------------------------------
      1 LINESTRING ( 70.00000000 130.00000000, 70.00000000 140.00000000,
80.00000000 140.00000000, 80.00000000 130.00000000, 70.00000000 130.00000000)
```

Referência Relacionada:

- "ST_ExteriorRing" na página 388
- "ST_NumInteriorRing" na página 459

ST_Intersection

ST_Intersection utiliza duas geometrias como parâmetros de entrada e retorna a geometria que é a interseção das duas geometrias especificadas. A interseção é a parte comum da primeira geometria e da segunda geometria. A geometria resultante é representada no sistema de referência espacial da primeira geometria.

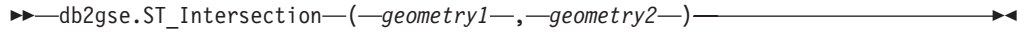
Se possível, o tipo específico da geometria retornada será ST_Point, ST_LineString ou ST_Polygon. Por exemplo, a interseção de um ponto e de um polígono é um ponto vazio ou único, representado como ST_MultiPoint.

Se qualquer uma das duas geometrias for nula, será retornado nulo.

Para dados não geodésicos, se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial. Para dados geodésicos, as duas geometrias devem estar no mesmo SRS (Spatial Reference System) geodésico.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry1
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a primeira geometria para calcular a interseção com *geometry2*.

geometry2
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a segunda geometria para calcular a interseção com *geometry1*.

Para dados geodésicos, as duas geometrias devem ser geodésicas e devem estar no mesmo SRS geodésico.

Tipo de Retorno:

db2gse.ST_Geometry

A dimensão da geometria retornada é a dimensão da entrada com a dimensão inferior, exceto para cadeias de linhas em dados geodésicos. Para dados geodésicos, a dimensão da interseção de duas cadeias de linhas é 0 (em outras palavras, a interseção é um ponto ou multiponto).

Exemplo:

Nos exemplos a seguir, os resultados foram reformatados para possibilitar a leitura. O espaçamento nos resultados irá variar de acordo com a exibição.

Este exemplo cria várias geometrias diferentes e depois determina a interseção (se houver) com a primeira.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('polygon((30 30, 30 50, 50 50, 50 30, 30 30))' ,0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('polygon((20 30, 30 30, 30 40, 20 40, 20 30))' ,0))

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('polygon((40 40, 40 60, 60 60, 60 40, 40 40))' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('linestring(60 60, 70 70)' ,0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('linestring(30 30, 60 60)' ,0))

SELECT a.id, b.id, CAST(ST_AsText(ST_Intersection(a.geometry, b.geometry))
   as VARCHAR(150)) Intersection
FROM sample_geoms a, sample_geoms b
WHERE a.id = 1
```

Resultados:

```
ID          ID          INTERSECTION
----------- ----------- --------------------------------------------------
          1           1 POLYGON (( 30.00000000 30.00000000, 50.00000000
30.00000000, 50.00000000 50.00000000, 30.00000000 50.00000000, 30.00000000
30.00000000))

          1           2 LINESTRING ( 30.00000000 40.00000000, 30.00000000
30.00000000)

          1           3 POLYGON (( 40.00000000 40.00000000, 50.00000000
40.00000000, 50.00000000 50.00000000, 40.00000000 50.00000000, 40.00000000
40.00000000))

          1           4 POINT EMPTY

          1           5 LINESTRING ( 30.00000000 30.00000000, 50.00000000
50.00000000)

  5 registros selecionados.
```

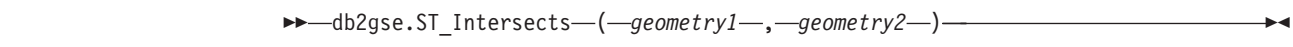
ST_Intersects

ST_Intersects utiliza duas geometrias como parâmetros de entrada e retorna 1 se as geometrias especificadas forem interseccionadas. Se as geometrias não se cruzarem, será retornado 0 (zero).

Se qualquer uma das duas geometrias for nula ou vazia, será retornado nulo.

Para dados não geodésicos, se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial. Para dados geodésicos, as duas geometrias devem estar no mesmo sistema de referência espacial (SRS) geodésico.

Sintaxe:



Parâmetro:

geometry1
Um valor de tipo ST_Geometry ou um de seus subtipos que representa a geometria para testar a interseção com *geometry2*.

geometry2
Um valor de tipo ST_Geometry ou um de seus subtipos que representa a geometria para testar a interseção com *geometry1*.

Restrições: Para dados geodésicos, as duas geometrias devem ser geodésicas e devem estar no mesmo SRS geodésico.

Tipo de Retorno:

INTEGER

Exemplo:

As seguintes instruções criam e ocupam as tabelas SAMPLE_GEOMETRIES1 e SAMPLE_GEOMETRIES2.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries1(id SMALLINT, spatial_type varchar(13),
    geometry ST_GEOMETRY);
CREATE TABLE sample_geometries2(id SMALLINT, spatial_type varchar(13),
    geometry ST_GEOMETRY);

INSERT INTO sample_geometries1(id, spatial_type, geometry)
VALUES
    ( 1, 'ST_Point', ST_Point('point(550 150)', 1) ),
    (10, 'ST_LineString', ST_LineString('linestring(800 800, 900 800)', 1)),
    (20, 'ST_Polygon', ST_Polygon('polygon((500 100, 500 200, 700 200,
          700 100, 500 100))', 1) )

INSERT INTO sample_geometries2(id, spatial_type, geometry)
VALUES
    (101, 'ST_Point', ST_Point('point(550 150)', 1) ),
    (102, 'ST_Point', ST_Point('point(650 200)', 1) ),
    (103, 'ST_Point', ST_Point('point(800 800)', 1) ),
    (110, 'ST_LineString', ST_LineString('linestring(850 250, 850 850)', 1)),
```

```
    (120, 'ST_Polygon', ST_Polygon('polygon((650 50, 650 150, 800 150,
           800 50, 650 50))', 1)),
    (121, 'ST_Polygon', ST_Polygon('polygon((20 20, 20 40, 40 40, 40 20,
           20 20))', 1) )
```

A seguinte instrução SELECT determina se as diversas geometrias nas tabelas SAMPLE_GEOMTRIES1 e SAMPLE_GEOMTRIES2 fazem interseção.

```
SELECT   sg1.id AS sg1_id, sg1.spatial_type AS sg1_type,
         sg2.id AS sg2_id, sg2.spatial_type AS sg2_type,
         CASE ST_Intersects(sg1.geometry, sg2.geometry)
            WHEN 0 THEN 'Geometries do not intersect'
            WHEN 1 THEN 'Geometries intersect'
         END AS intersects
FROM     sample_geometries1 sg1, sample_geometries2 sg2
ORDER BY sg1.id
```

Resultados:

```
SG1_ID SG1_TYPE      SG2_ID SG2_TYPE      INTERSECTS
------ ------------- ------ ------------- --------------------------
     1 ST_Point         101 ST_Point      Geometries intersect
     1 ST_Point         102 ST_Point      Geometries do not intersect
     1 ST_Point         103 ST_Point      Geometries do not intersect
     1 ST_Point         110 ST_LineString Geometries do not intersect
     1 ST_Point         120 ST_Polygon    Geometries do not intersect
     1 ST_Point         121 ST_Polygon    Geometries do not intersect
    10 ST_LineString    101 ST_Point      Geometries do not intersect
    10 ST_LineString    102 ST_Point      Geometries do not intersect
    10 ST_LineString    103 ST_Point      Geometries intersect
    10 ST_LineString    110 ST_LineString Geometries intersect
    10 ST_LineString    120 ST_Polygon    Geometries do not intersect
    10 ST_LineString    121 ST_Polygon    Geometries do not intersect
    20 ST_Polygon       101 ST_Point      Geometries intersect
    20 ST_Polygon       102 ST_Point      Geometries intersect
    20 ST_Polygon       103 ST_Point      Geometries do not intersect
    20 ST_Polygon       110 ST_LineString Geometries do not intersect
    20 ST_Polygon       120 ST_Polygon    Geometries intersect
    20 ST_Polygon       121 ST_Polygon    Geometries do not intersect
```

Referência Relacionada:

• "Funções que Comparam Recursos Geográficos" na página 308

ST_Is3d

ST_Is3d utiliza uma geometria como um parâmetro de entrada e retorna 1 se a geometria especificada tiver coordenadas Z. Caso contrário, será retornado 0 (zero).

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada quanto à existência de coordenadas Z.

Tipo de Retorno:

INTEGER

Exemplo:

Neste exemplo, são criadas várias geometrias com e sem coordenadas Z e coordenadas M (medidas). ST_Is3d é utilizada para determinar qual delas contém coordenadas Z.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('point EMPTY',0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('polygon((40 120, 90 120, 90 150, 40 150, 40 120))' ,0))

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('multipoint m (10 10 5, 50 10 6, 10 30 8)' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('linestring z (10 10 166, 20 10 168)',0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('point zm (10 10 16 30)' ,0))


SELECT id, ST_Is3d(geometry)  Is_3D
FROM sample_geoms
```

Resultados:

```
ID          IS_3D
----------- -----------
          1           0
          2           0
          3           0
          4           1
          5           1
```

ST_IsClosed

ST_IsClosed utiliza uma curva ou multicurva como parâmetro de entrada e retorna 1 se a curva ou multicurva especificada for fechada. Caso contrário, será retornado 0 (zero).

Uma curva é fechada se o ponto de início e o ponto de término forem iguais. Se a curva tiver coordenadas Z, elas deverão ser iguais para os pontos de início e de término. Caso contrário, os pontos não são considerados iguais e a curva não é fechada. Uma multicurva é fechada se cada uma de suas curvas for fechada.

Se a curva ou multicurva especificada for vazia, será retornado 0 (zero). Se for nula, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

curve Um valor do tipo ST_Curve ou ST_MultiCurve ou um de seus subtipos que representa a curva ou multicurva que será testada.

Tipo de Retorno:

INTEGER

Exemplos:

Exemplo 1:

Este exemplo cria várias cadeias de linhas. As duas últimas cadeias de linhas têm as mesmas coordenadas X e Y, mas uma cadeia de linha contém coordenadas Z variantes que impedem o fechamento da cadeia de linhas e a outra cadeia de linhas contém coordenas M variantes (medidas) que não afetam o fechamento da cadeia de linhas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, geometry ST_Linestring)

INSERT INTO sample_lines VALUES
       (1, ST_Linestring('linestring EMPTY',0))

INSERT INTO sample_lines VALUES
       (2, ST_Linestring('linestring(10 10, 20 10, 20 20)' ,0))

INSERT INTO sample_lines VALUES
       (3, ST_Linestring('linestring(10 10, 20 10, 20 20, 10 10)' ,0))

INSERT INTO sample_lines VALUES
       (4, ST_Linestring('linestring m(10 10 1, 20 10 2, 20 20 3,
        10 10 4)' ,0))

INSERT INTO sample_lines VALUES
       (5, ST_Linestring('linestring z(10 10 5, 20 10 6, 20 20 7,
        10 10 8)' ,0))

SELECT id, ST_IsClosed(geometry)  Is_Closed
FROM sample_lines
```

Resultados:

```
ID          IS_CLOSED
----------- -----------
          1           0
          2           0
          3           1
          4           1
          5           0
```

Exemplo 2:

Neste exemplo, são criadas duas cadeias multilinha. ST_IsClosed é utilizado para determinar se as cadeias multilinha são fechadas. A primeira não é fechada, mesmo que todas as curvas juntas formem um loop completo fechado. Isto ocorre porque cada curva por si não é fechada.

A segunda cadeia multilinha é fechada porque cada curva por si é fechada.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mlines (id INTEGER, geometry ST_MultiLinestring)
INSERT INTO sample_mlines VALUES
       (6, ST_MultiLinestring('multilinestring((10 10, 20 10, 20 20),
                                               (20 20, 30 20, 30 30),
                                               (30 30, 10 30, 10 10))',0))

INSERT INTO sample_mlines VALUES
       (7, ST_MultiLinestring('multilinestring((10 10, 20 10, 20 20, 10 10 ),
                                                (30 30, 50 30, 50 50,
                                                 30 30 ))',0))

SELECT id, ST_IsClosed(geometry)  Is_Closed
FROM sample_mlines
```

Resultados:

```
ID          IS_CLOSED
----------- -----------
          6           0
          7           1
```

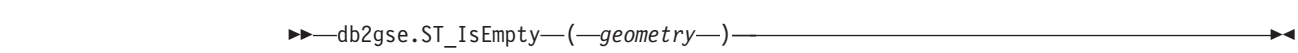
ST_IsEmpty

ST_IsEmpty utiliza uma geometria como um parâmetro de entrada e retorna 1 se a geometria especificada for vazia. Caso contrário, será retornado 0 (zero). Uma geometria é vazia se não tiver pontos que a definam.

Se a geometria indicada for nula, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada.

Tipo de Retorno:

INTEGER

Exemplo:

O código a seguir cria três geometrias e determina se são vazias.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('point EMPTY',0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('polygon((40 120, 90 120, 90 150, 40 150, 40 120))' ,0))
```

```
INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('multipoint m (10 10 5, 50 10 6, 10 30 8)' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('linestring z (10 10 166, 20 10 168)',0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('point zm (10 10 16 30)' ,0))


SELECT id, ST_IsEmpty(geometry)  Is_Empty
FROM sample_geoms
```

Resultados:

```
ID          IS_EMPTY
----------- -----------
          1           1
          2           0
          3           0
          4           0
          5           0
```

ST_IsMeasured

ST_IsMeasured utiliza uma geometria como um parâmetro de entrada e retorna 1 se a geometria especificada tiver coordenadas M (medidas). Caso contrário, será retornado 0 (zero).

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria a ser testada quanto à existência de coordenadas M (medidas).

Tipo de Retorno:

INTEGER

Exemplo:

Neste exemplo, são criadas várias geometrias com e sem coordenadas Z e coordenadas M (medidas). ST_IsMeasured é utilizada para determinar qual delas contém medidas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('point EMPTY',0))
```

```
INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('polygon((40 120, 90 120, 90 150, 40 150, 40 120))' ,0))

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('multipoint m (10 10 5, 50 10 6, 10 30 8)' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('linestring z (10 10 166, 20 10 168)',0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('point zm (10 10 16 30)' ,0))


SELECT id, ST_IsMeasured(geometry)  Is_Measured
FROM sample_geoms
```

Resultados:

```
ID          IS_MEASURED
----------- -----------
          1           0
          2           0
          3           1
          4           0
          5           1
```

ST_IsRing

ST_IsRing utiliza uma curva como um parâmetro de entrada e retorna 1 se for um anel. Caso contrário, será retornado 0 (zero). Uma curva é um anel se for simples e fechada.

Se a curva especificada for vazia, então será retornado 0 (zero). Se for nula, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

curve Um valor do tipo ST_Curve ou um de seus subtipos que representa a curva a ser testada.

Tipo de Retorno:

INTEGER

Exemplos:

Neste exemplo, são criadas quatro cadeias de linhas. ST_IsRing é utilizado para verificar se elas são anéis. A última não é considerada um anel mesmo que seja fechada, porque o caminho cruza sobre ela própria.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, geometry ST_Linestring)

INSERT INTO sample_lines VALUES
       (1, ST_Linestring('linestring EMPTY',0))
```

```
INSERT INTO sample_lines VALUES
       (2, ST_Linestring('linestring(10 10, 20 10, 20 20)' ,0))

INSERT INTO sample_lines VALUES
       (3, ST_Linestring('linestring(10 10, 20 10, 20 20, 10 10)' ,0))

INSERT INTO sample_lines VALUES
       (4, ST_Linestring('linestring(10 10, 20 10, 10 20, 20 20, 10 10)' ,0))


SELECT id, ST_IsClosed(geometry) Is_Closed, ST_IsRing(geometry)  Is_Ring
FROM sample_lines
```

Resultados:

```
ID          IS_CLOSED   IS_RING
----------- ----------- -----------
          1           1           0
          2           0           0
          3           1           1
          4           1           0
```

Referência Relacionada:

- "ST_IsClosed" na página 411
- "ST_IsSimple" na página 416

ST_IsSimple

ST_IsSimple utiliza uma geometria como um parâmetro de entrada e retorna 1 se a geometria especificada for simples. Caso contrário, será retornado 0 (zero).

Pontos, superfícies e multisuperfícies são sempre simples. Uma curva é simples se não passar duas vezes pelo mesmo ponto; um multiponto é simples se não tiver dois pontos iguais; e uma multicurva é simples se todas as suas curvas forem simples e as únicas interseções ocorrerem em pontos que estão no limite das curvas na multicurva.

Se a geometria especificada for vazia, será retornado o valor 1. Se for nula, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada.

Tipo de Retorno:

INTEGER

Exemplos:

Neste exemplo, são criadas várias geometrias e é verificado se elas são simples. A geometria com um ID 4 não é considerada simples porque ela contém mais de um ponto que é igual. A geometria com um ID 6 não é considerada simples porque a cadeia de linhas cruza com ela mesma.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('point EMPTY' ,0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('point (21 33)' ,0))

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('multipoint(10 10, 20 20, 30 30)' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('multipoint(10 10, 20 20, 30 30, 20 20)' ,0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('linestring(60 60, 70 60, 70 70)' ,0))

INSERT INTO sample_geoms VALUES
       (6, ST_Geometry('linestring(20 20, 30 30, 30 20, 20 30  )' ,0))

INSERT INTO sample_geoms VALUES
       (7, ST_Geometry('polygon((40 40, 50 40, 50 50, 40 40  ))' ,0))

SELECT id, ST_IsSimple(geometry) Is_Simple
FROM sample_geoms
```

Resultados:

```
ID          IS_SIMPLE
----------- -----------
          1           1
          2           1
          3           1
          4           0
          5           1
          6           0
          7           1
```

ST_IsValid

ST_IsValid utiliza uma geometria como um parâmetro de entrada e retorna 1 se for válida. Caso contrário, será retornado 0 (zero). Uma geometria somente será válida se todos os atributos no tipo estruturado forem consistentes com a representação interna de dados da geometria e se a representação interna não estiver danificada.

Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos.

Tipo de Retorno:

INTEGER

Exemplo:

Este exemplo cria várias geometrias e utiliza ST_IsValid para verificar se elas são válidas. Todas as geometrias são válidas porque as rotinas do construtor, como ST_Geometry, não permitem a construção de geometrias inválidas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms VALUES
       (1, ST_Geometry('point EMPTY',0))

INSERT INTO sample_geoms VALUES
       (2, ST_Geometry('polygon((40 120, 90 120, 90 150, 40 150, 40 120))' ,0))

INSERT INTO sample_geoms VALUES
       (3, ST_Geometry('multipoint m (10 10 5, 50 10 6, 10 30 8)' ,0))

INSERT INTO sample_geoms VALUES
       (4, ST_Geometry('linestring z (10 10 166, 20 10 168)',0))

INSERT INTO sample_geoms VALUES
       (5, ST_Geometry('point zm (10 10 16 30)' ,0))


SELECT id, ST_IsValid(geometry)  Is_Valid
FROM sample_geoms
```

Resultados:

```
ID          IS_VALID
----------- -----------
          1           1
          2           1
          3           1
          4           1
          5           1
```

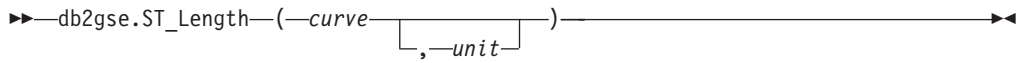
ST_Length

ST_Length utiliza uma curva ou multicurva e, opcionalmente, uma unidade como parâmetros de entrada e retorna o comprimento da curva ou multicurva especificada na unidade de medida padrão ou fornecida.

Se a curva ou multicurva especificada for nula ou vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

curve Um valor do tipo ST_Curve ou ST_MultiCurve que representa as curvas para as quais o comprimento é retornado.

unit Um valor VARCHAR(128) que identifica as unidades nas quais o comprimento da curva é medido. As unidades de medida suportadas estão listadas na exibição do catálogo DB2GSE.ST_UNITS_OF_MEASURE.

Se o parâmetro *unit* for omitido, as seguintes regras serão utilizadas para determinar a unidade na qual o comprimento é medido:

- Se *curve* estiver em um sistema de coordenadas projetadas ou geocêntricas, a unidade linear associada a este sistema de coordenadas será a padrão.
- Se *curve* estiver em um sistema de coordenadas geográficas, mas não estiver em um SRS (Spatial Reference System) geodésico, a unidade angular associada a este sistema de coordenadas será a padrão.
- Se *curve* estiver em um SRS geodésico, a unidade de medida padrão será em metros.

Restrições para conversões de unidades: Será retornado um erro (SQLSTATE 38SU4) se ocorrer qualquer uma das condições a seguir:

- A *curve* está em um sistema de coordenadas não especificado e o parâmetro *unit* é especificado.
- A *curve* está em um sistema de coordenadas projetadas e uma unidade angular é especificada.
- A *curve* está em um sistema de coordenadas geográficas, mas não está em um SRS geodésico e uma unidade linear é especificada.
- A *curve* está em um SRS geodésico e uma unidade angular é especificada.

Tipo de Retorno:

DOUBLE

Exemplos:

As instruções SQL a seguir criam uma tabela SAMPLE_GEOMETRIES e inserem uma linha e uma multilinha na tabela.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_geometries(id SMALLINT, spatial_type varchar(20),
    geometry ST_GEOMETRY)

INSERT INTO sample_geometries(id, spatial_type, geometry)
VALUES
    (1110, 'ST_LineString', ST_LineString('linestring(50 10, 50 20)', 1)),
    (1111, 'ST_MultiLineString', ST_MultiLineString('multilinestring
                            ((33 2, 34 3, 35 6),
                            (28 4, 29 5, 31 8, 43 12),
                            (39 3, 37 4, 36 7))', 1))
```

Exemplo 1:

A instrução SELECT a seguir calcula o comprimento da linha na tabela SAMPLE_GEOMTRIES.

```
SELECT id, spatial_type, cast(ST_Length(geometry..ST_ToLineString)
   AS DECIMAL(7, 2)) AS "Line Length"
FROM   sample_geometries
WHERE  id = 1110
```

Resultados:

```
ID     SPATIAL_TYPE         Line Length
------ -------------------- -----------
  1110 ST_LineString              10.00
```

Exemplo 2:

A instrução SELECT a seguir calcula o comprimento da multilinha na tabela SAMPLE_GEOMTRIES.

```
SELECT id, spatial_type, ST_Length(ST_ToMultiLine(geometry))
   AS multiline_length
FROM   sample_geometries
WHERE  id = 1111
```

Resultados:

```
ID     SPATIAL_TYPE         MULTILINE_LENGTH
 ------ -------------------- ------------------------
   1111 ST_MultiLineString     +2.76437123387202E+001
```

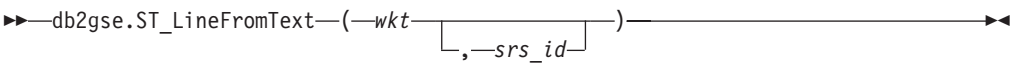
ST_LineFromText

ST_LineFromText utiliza uma representação de texto reconhecida de uma cadeia de linhas e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna a cadeia de linhas correspondente.

Se a representação de texto reconhecida for nula, então nulo é retornado.

A versão preferida para esta funcionalidade é ST_LineString.

Sintaxe:



Parâmetro:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida da cadeia de linhas resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da cadeia de linhas resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identifica um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então um erro é retornado (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_LineString

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir utiliza a função ST_LineFromText para criar e inserir uma linha a partir de uma representação de linha de WKT (Well-Known Text). As linhas são inseridas na tabela SAMPLE_LINES com um ID e um valor de linha no sistema de referência espacial 1 na representação WKT.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_lines(id SMALLINT, geometry ST_LineString)

INSERT INTO sample_lines (id, geometry)
VALUES
    (1110, ST_LineFromText('linestring(850 250, 850 850)', 1) ),
    (1111, ST_LineFromText('linestring empty', 1) )

SELECT id, cast(geometry..ST_AsText AS varchar(75)) AS linestring
  FROM sample_lines
```

Resultados:

```
ID     LINESTRING
------ ---------------------------------------------------------------
  1110 LINESTRING ( 850.00000000 250.00000000, 850.00000000 850.00000000)
  1111 LINESTRING EMPTY
```

Referência Relacionada:

- "Representação WKT (Well-Known Text)" na página 519

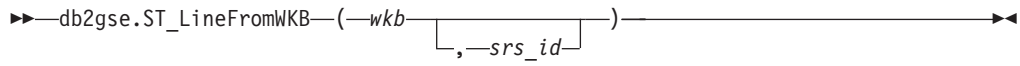
ST_LineFromWKB

ST_LineFromWKB utiliza uma representação binária reconhecida de uma cadeia de linhas e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna a cadeia de linhas correspondente.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

A versão preferida para esta personalidade é ST_LineString.

Sintaxe:



Parâmetro:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida da cadeia de linhas resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da cadeia de linhas resultante.

Se o parâmetro ***srs_id*** for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identifica um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então um erro é retornado (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_LineString

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

O código a seguir utiliza a função ST_LineFromWKB para criar e inserir uma linha a partir de uma representação binária reconhecida. A linha é inserida na tabela SAMPLE_LINES com um ID e uma linha no sistema de referência espacial 1 na representação WKB.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_lines(id SMALLINT, geometry ST_LineString, wkb BLOB(32k))

INSERT INTO sample_lines (id, geometry)
VALUES
    (1901, ST_LineString('linestring(850 250, 850 850)', 1) ),
    (1902, ST_LineString('linestring(33 2, 34 3, 35 6)', 1) )

UPDATE sample_lines AS temp_correlated
SET    wkb = geometry..ST_AsBinary
WHERE  id = temp_correlated.id

SELECT id, cast(ST_LineFromWKB(wkb)..ST_AsText AS varchar(90)) AS line
  FROM sample_lines
```

Resultados:

```
ID     LINE
------ ------------------------------------------------------------------
  1901 LINESTRING ( 850.00000000 250.00000000, 850.00000000 850.00000000)

  1902 LINESTRING ( 33.00000000 2.00000000, 34.00000000 3.00000000,
35.00000000 6.00000000)
```

Referência Relacionada:

• "Representação WKB (Well-Known Binary)" na página 524

ST_LineString

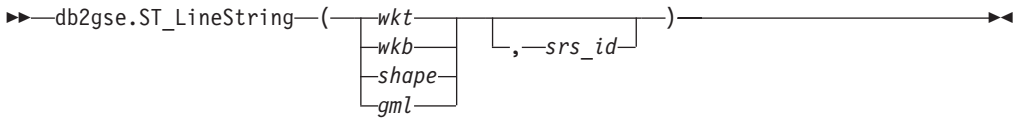
ST_LineString constrói uma cadeia de linhas a partir de uma das seguintes entradas:

- Uma representação de texto reconhecida
- Uma representação binária reconhecida
- Uma representação de formatos ESRI
- Uma representação na GML (Geography Markup Language)

Um identificador do sistema de referência espacial pode ser fornecido opcionalmente para identificar o sistema de referência espacial no qual a cadeia de linhas resultante está localizada.

Se a representação de texto reconhecida, a representação binária reconhecida, a representação de formatos ESRI ou a representação GML for nula, será retornado nulo.

Sintaxe:



Parâmetro:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do polígono resultante.

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do polígono resultante.

shape Um valor do tipo BLOB(2G) que representa a representação de formatos ESRI do polígono resultante.

gml Um valor do tipo CLOB(2G) que representa o polígono resultante utilizando a GML (Geography Markup Language).

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do polígono resultante.

Se o parâmetro *srs_id* for omitido, será utilizado o sistema de referência espacial com o identificador numérico 0 (zero).

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, será retornado um erro (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_LineString

Exemplos:

O código a seguir utiliza a função ST_LineString para criar e inserir uma linha a partir de uma representação de linha de WKT (Well-Known Text) ou a partir de uma representação WKB (Well-Known Binary).

O exemplo a seguir insere uma linha na tabela SAMPLE_LINES com um ID e linha no sistema de referência espacial 1 nas representações WKT e GML

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_lines(id SMALLINT, geometry ST_LineString)

INSERT INTO sample_lines (id, geometry)
VALUES
    (1110, ST_LineString('linestring(850 250, 850 850)', 1) ),
    (1111, ST_LineString('<gml:LineString srsName=";EPSG:4269";><gml:coord>
                          <gml:X>90</gml:X><gml:Y>90</gml:Y>
                          </gml:coord><gml:coord><gml:X>100</gml:X>
                          <gml:Y>100</gml:Y></gml:coord>
                          </gml:LineString>', 1) )

SELECT id, cast(geometry..ST_AsText AS varchar(75)) AS linestring
FROM   sample_lines
```

Resultados:

```
ID     LINESTRING
------ ------------------------------------------------------------------
  1110 LINESTRING ( 850.00000000 250.00000000, 850.00000000 850.00000000)
  1111 LINESTRING ( 90.00000000 90.00000000, 100.00000000 100.00000000)
```

Referência Relacionada:

- "Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados" na página 299
- "Representação WKT (Well-Known Text)" na página 519

ST_LineStringN

ST_LineStringN utiliza uma cadeia multilinha e um índice como parâmetros de entrada e retorna uma cadeia de linhas que é identificada pelo índice. A cadeia de linhas resultante é representada no sistema de referência espacial da cadeia multilinha especificada.

Se a cadeia multilinha especificada for nula ou vazia, ou se o índice for menor do que 1 ou maior do que o número de cadeias de linhas, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

multi_linestring
: Um valor do tipo ST_MultiLineString que representa a cadeia multilinha a partir da qual a cadeia de linhas identificada por *index* é retornada.

index
: Um valor do tipo INTEGER que identifica a *ú*ltima cadeia de linhas, que deve ser retornada de *multi_linestring*.

  Se *index* for menor do que 1 ou maior do que o número de cadeias de linhas em *multi_linestring*, será retornado nulo e também uma condição de aviso (SQLSTATE 01HS0).

Tipo de Retorno:

db2gse.ST_LineString

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

A instrução SELECT ilustra como escolher a segunda geometria dentro de uma cadeia multilinha na tabela SAMPLE_MLINES.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

CREATE TABLE sample_mlines (id INTEGER,
  geometry ST_MULTILINESTRING)

INSERT INTO sample_mlines(id, geometry)
VALUES
```

```
    (1110, ST_MultiLineString('multilinestring
                              ((33 2, 34 3, 35 6),
                              (28 4, 29 5, 31 8, 43 12),
                              (39 3, 37 4, 36 7))', 1) ),
    (1111, ST_MLineFromText('multilinestring(
                            (61 2, 64 3, 65 6),
                            (58 4, 59 5, 61 8),
                            (69 3, 67 4, 66 7, 68 9))', 1) )

SELECT id, cast(ST_LineStringN(geometry, 2)..ST_AsText
  AS varchar(110)) AS second_linestring
FROM   sample_mlines
```

Resultados:

```
ID          SECOND_LINESTRING
----------- ------------------------------------------------------------------
1110         LINESTRING ( 28.00000000 4.00000000, 29.00000000
   5.00000000, 31.00000000 8.00000000, 43.00000000 12.00000000)

1111         LINESTRING ( 58.00000000 4.00000000, 59.00000000
   5.00000000, 61.00000000 8.00000000)
```

Referência Relacionada:

- "ST_NumLineStrings" na página 460

ST_M

ST_M pode:

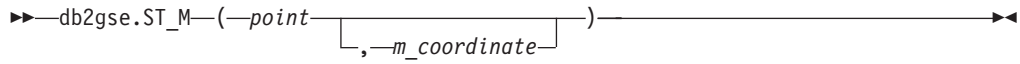
- Utilizar um ponto como parâmetro de entrada e retornar sua coordenada M (medida)
- Utilizar um ponto e uma coordenada M e retornar o próprio ponto com sua coordenada M definida como a medida especificada, mesmo que o ponto especificado não tenha nenhuma coordenada M existente.

Se a coordenada M especificada for nula, a coordenada M do ponto será removida.

Se o ponto especificado for nulo ou vazio, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

point Um valor do tipo ST_Point para o qual a coordenada M é retornada ou modificada.

m_coordinate
Um valor do tipo DOUBLE que representa a nova coordenada M para *point*.

Se *m_coordinate* for nulo, a coordenada M será removida de *point*.

Tipos de Retorno:

- DOUBLE, se *m_coordinate* não for especificado

- db2gse.ST_Point, se *m_coordinate* for especificado

Exemplos:

Estes exemplos ilustram a utilização da função ST_M. Três pontos são criados e inseridos na tabela SAMPLE_POINTS. Todos eles estão no sistema de referência espacial que tem um ID 1.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)

INSERT INTO sample_points
  VALUES (1, ST_Point (2, 3, 32, 5, 1))

INSERT INTO sample_points
  VALUES (2, ST_Point (4, 5, 20, 4, 1))

INSERT INTO sample_points
  VALUES (3, ST_Point (3, 8, 23, 7, 1))
```

Exemplo 1:

Este exemplo encontra a coordenada M dos pontos na tabela SAMPLE_POINTS.

```
SELECT id, ST_M (geometry) M_COORD
  FROM sample_points
```

Resultados:

```
ID          M_COORD
----------- ------------------------
          1 +5.00000000000000E+000
          2 +4.00000000000000E+000
          3 +7.00000000000000E+000
```

Exemplo 2:

Este exemplo retorna um dos pontos com sua coordenada M definida como 40.

```
SELECT id, CAST (ST_AsText (ST_M (geometry, 40) )
  AS VARCHAR(60) ) M_COORD_40
  FROM sample_points
  WHERE id=3
```

Resultados:

```
ID         M_COORD_40
---------- ------------------------------------------------------
         3 POINT ZM (3.00000000 8.00000000 23.00000000 40.00000000)
```

Referência Relacionada:

- "ST_X" na página 506
- "ST_Y" na página 507
- "ST_Z" na página 509

ST_MaxM

ST_MaxM utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada M máxima.

Se a geometria especificada for nula ou vazia, ou se não tiver coordenadas M, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos para o qual a coordenada M máxima é retornada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_MaxM. Três polígonos são criados e inseridos na tabela SAMPLE_POLYS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon zm ((110 120 20 3,
                                      110 140 22 3,
                                      120 130 26 4,
                                      110 120 20 3))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon zm ((0 0 40 7,
                                      0 4 35 9,
                                      5 4 32 12,
                                      5 0 31 5,
                                      0 0 40 7))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon('polygon zm ((12 13 10 16,
                                      8 4 10 12,
                                      9 4 12 11,
                                      12 13 10 16))', 0) )
```

Exemplo 1:

Este exemplo encontra a coordena M máxima de cada polígono em SAMPLE_POLYS.

```
SELECT id, CAST ( ST_MaxM(geometry) AS INTEGER) MAX_M
  FROM sample_polys
```

Resultados:

```
ID          MAX_M
----------- -----------
          1           4
          2          12
          3          16
```

Exemplo 2:

Este exemplo encontra a coordenada M máxima existente para todos os polígonos na coluna GEOMETRY.

```
SELECT CAST ( MAX ( ST_MaxM(geometry) ) AS INTEGER) OVERALL_MAX_M
  FROM sample_polys
```

Resultados:

```
OVERALL_MAX_M
-------------
           16
```

Conceitos Relacionados:

- "ST_MaxX" na página 428

Referência Relacionada:

- "ST_MaxY" na página 430
- "ST_MaxZ" na página 431
- "ST_MinM" na página 438

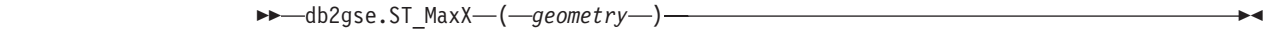
ST_MaxX

ST_MaxX utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada X máxima.

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos para o qual a coordenada X máxima é retornada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_MaxX. Três polígonos são criados e inseridos na tabela SAMPLE_POLYS. O terceiro exemplo ilustra como você pode utilizar todas as funções que retornam os valores de coordenas máxima e mínima para avaliar o intervalo espacial das geometrias que podem ser armazenadas em uma determinada coluna espacial.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon zm ((110 120 20 3,
                                      110 140 22 3,
                                      120 130 26 4,
                                      110 120 20 3))', 0) )
```

```
INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon zm ((0 0 40 7,
                                      0 4 35 9,
                                      5 4 32 12,
                                      5 0 31 5,
                                      0 0 40 7))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon('polygon zm ((12 13 10 16,
                                      8 4 10 12,
                                      9 4 12 11,
                                      12 13 10 16))', 0) )
```

Exemplo 1:

Este exemplo encontra a coordena X máxima de cada polígono em SAMPLE_POLYS.

```
SELECT id, CAST ( ST_MaxX(geometry) AS INTEGER) MAX_X_COORD
  FROM sample_polys
```

Resultados:

```
ID          MAX_X_COORD
----------- ------------
          1          120
          2            5
          3           12
```

Exemplo 2:

Este exemplo encontra a coordenada X máxima existente para todos os polígonos na coluna GEOMETRY.

```
SELECT CAST ( MAX ( ST_MaxX(geometry) ) AS INTEGER) OVERALL_MAX_X
  FROM sample_polys
```

Resultados:

```
OVERALL_MAX_X
-------------
          120
```

Exemplo 3:

Este exemplo encontra a extensão espacial (mínima total e máxima total) de todos os polígonos na tabela SAMPLE_POLYS. Este cálculo geralmente é utilizado para comparar a extensão espacial real das geometrias com a extensão do sistema de referência espacial associado aos dados para determinar se os dados podem ser expandidos.

```
SELECT CAST ( MIN (ST_MinX (geometry)) AS INTEGER) MIN_X,
       CAST ( MIN (ST_MinY (geometry)) AS INTEGER) MIN_Y,
       CAST ( MIN (ST_MinZ (geometry)) AS INTEGER) MIN_Z,
       CAST ( MIN (ST_MinM (geometry)) AS INTEGER) MIN_M,
       CAST ( MAX (ST_MaxX (geometry)) AS INTEGER) MAX_X,
       CAST ( MAX (ST_MaxY (geometry)) AS INTEGER) MAX_Y,
       CAST ( MAX (ST_MaxZ (geometry)) AS INTEGER) MAX_Z,
       CAST ( MAX (ST_MaxmM(geometry)) AS INTEGER) MAX_M,
  FROM sample_polys
```

Resultados:

```
MIN_X     MIN_Y     MIN_Z     MIN_M     MAX_X     MAX_Y     MAX_Z     MAX_M
--------- --------- --------- --------- --------- --------- --------- ----------
        0         0        10         3       120       140        40         16
```

Referência Relacionada:

- "ST_MaxM" na página 426
- "ST_MaxY" na página 430
- "ST_MaxZ" na página 431
- "ST_MinX" na página 439

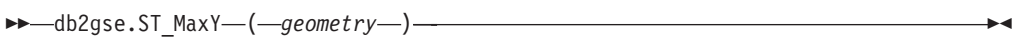
ST_MaxY

ST_MaxY utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada Y máxima.

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos para o qual a coordenada Y máxima é retornada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_MaxY. Três polígonos são criados e inseridos na tabela SAMPLE_POLYS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon zm ((110 120 20 3,
                                      110 140 22 3,
                                      120 130 26 4,
                                      110 120 20 3))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon zm ((0 0 40 7,
                                      0 4 35 9,
                                      5 4 32 12,
                                      5 0 31 5,
                                      0 0 40 7))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon('polygon zm ((12 13 10 16,
                                      8 4 10 12,
                                      9 4 12 11,
                                      12 13 10 16))', 0) )
```

Exemplo 1:

Este exemplo encontra a coordena Y máxima de cada polígono em SAMPLE_POLYS.

```
SELECT id, CAST ( ST_MaxY(geometry) AS INTEGER) MAX_Y
  FROM sample_polys
```

Resultados:

```
ID          MAX_Y
----------- ------------
          1          140
          2            4
          3           13
```

Exemplo 2:

Este exemplo encontra a coordenada Y máxima existente para todos os polígonos na coluna GEOMETRY.

```
SELECT CAST ( MAX ( ST_MaxY(geometry) ) AS INTEGER) OVERALL_MAX_Y
  FROM sample_polys
```

Resultados:

```
OVERALL_MAX_Y
--------------
          140
```

Conceitos Relacionados:

- "ST_MaxX" na página 428

Referência Relacionada:

- "ST_MaxM" na página 426
- "ST_MaxZ" na página 431
- "ST_MinY" na página 441

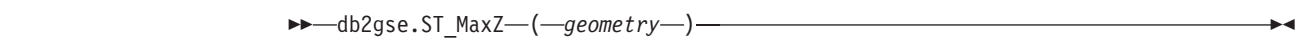
ST_MaxZ

ST_MaxZ utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada Z máxima.

Se a geometria especificada for nula ou vazia, ou se não tiver coordenadas Z, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos para o qual a coordenada Z máxima é retornada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_MaxZ. Três polígonos são criados e inseridos na tabela SAMPLE_POLYS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon zm ((110 120 20 3,
                                      110 140 22 3,
                                      120 130 26 4,
                                      110 120 20 3))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon zm ((0 0 40 7,
                                      0 4 35 9,
                                      5 4 32 12,
                                      5 0 31 5,
                                      0 0 40 7))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon('polygon zm ((12 13 10 16,
                                      8 4 10 12,
                                      9 4 12 11,
                                      12 13 10 16))', 0) )
```

Exemplo 1:

Este exemplo encontra a coordena Z máxima de cada polígono em SAMPLE_POLYS.

```
SELECT id, CAST ( ST_MaxZ(geometry) AS INTEGER) MAX_Z
  FROM sample_polys
```

Resultados:

```
ID          MAX_Z
----------- -----------
          1          26
          2          40
          3          12
```

Exemplo 2:

Este exemplo encontra a coordenada Z máxima existente para todos os polígonos na coluna GEOMETRY.

```
SELECT CAST ( MAX ( ST_MaxZ(geometry) ) AS INTEGER) OVERALL_MAX_Z
  FROM sample_polys
```

Resultados:

```
OVERALL_MAX_Z
-------------
           40
```

Conceitos Relacionados:

- "ST_MaxX" na página 428

Referência Relacionada:

- "ST_MaxM" na página 426

- "ST_MaxY" na página 430
- "ST_MinZ" na página 442

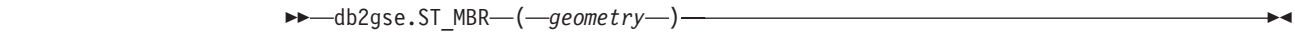
ST_MBR

ST_MBR utiliza uma geometria como um parâmetro de entrada e retorna seu retângulo limitador mínimo.

Se a geometria especificada for um ponto, então o próprio ponto será retornado. Se a geometria for uma cadeia de linhas horizontal ou uma cadeia de linhas vertical e o sistema de referência espacial não for geodésico, a própria cadeia de linhas horizontal ou vertical será retornada. Caso contrário, o retângulo limitador mínimo da geometria será retornado como um polígono. Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria à qual o retângulo de limite mínimo é retornado.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

Este exemplo ilustra como a função ST_MBR pode ser utilizada para retornar o retângulo de limite mínimo de um polígono. Como a geometria especificada é um polígono, o retângulo de limite mínimo é retornado como um polígono.

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente aqui para possibilitar a leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon ('polygon (( 5 5, 7 7, 5 9, 7 9, 9 11, 13 9,
                                     15 9, 13 7, 15 5, 9 6, 5 5))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon ('polygon (( 20 30, 25 35, 30 30, 20 30))', 0) )

SELECT id, CAST (ST_AsText ( ST_MBR(geometry)) AS VARCHAR(150) ) MBR
FROM sample_polys
```

Resultados:

```
ID        MBR
--------- ------------------------------------------------------------
        1 POLYGON (( 5.00000000 5.00000000, 15.00000000 5.00000000,
```

```
     15.00000000 11.00000000, 5.00000000 11.00000000,
      5.00000000 5.00000000))
2 POLYGON (( 20.00000000 30.00000000, 30.00000000 30.00000000,
     30.00000000 35.00000000, 20.00000000 35.00000000,
     20.00000000 30.00000000 ))
```

Referência Relacionada:

- "ST_Envelope" na página 381
- "ST_MBRIntersects" na página 434

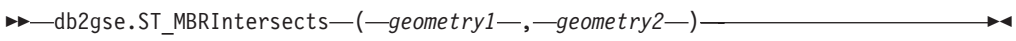
ST_MBRIntersects

ST_MBRIntersects utiliza duas geometrias como parâmetros de entrada e retorna 1 se os retângulos de limite mínimo das duas geometrias fizerem interseção. Caso contrário, será retornado 0 (zero). O retângulo de limite mínimo de um ponto e uma cadeia de linhas horizontal ou vertical representa a própria geometria.

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Se qualquer uma das geometrias especificadas for nula ou vazia, será retornado nulo.

Sintaxe:



Parâmetros:

geometry1
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria cujo retângulo de limite mínimo será testado para interseção com o retângulo de limite mínimo de *geometry2*.

geometry2
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria cujo retângulo de limite mínimo será testado para interseção com o retângulo de limite mínimo de *geometry1*.

Tipo de Retorno:

INTEGER

Exemplos:

Estes exemplos ilustram a utilização de ST_MBRIntersects para saber se dois polígonos que não fazem interseção estão próximos, verificando se seus retângulos de limite mínimo fazem interseção. O primeiro exemplo utiliza a expressão SQL CASE. O segundo exemplo utiliza uma única instrução SELECT para localizar os polígonos que fazem interseção com o retângulo de limite mínimo do polígono com ID = 2.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon ('polygon (( 0 0, 30 0, 40 30, 40 35,
```

```
                                    5 35, 5 10, 20 10, 20 5, 0 0 ))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon ('polygon (( 15 15, 15 20, 60 20, 60 15,
                                    15 15 ))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon ('polygon (( 115 15, 115 20, 160 20, 160 15,
                                    115 15 ))', 0) )
```

Exemplo 1:

A instrução SELECT a seguir utiliza uma expressão CASE para localizar os IDs dos polígonos que têm retângulos de limite mínimo que fazem interseção.

```
SELECT a.id, b.id,
    CASE ST_MBRIntersects (a.geometry, b.geometry)
      WHEN 0 THEN 'MBRs do not intersect'
      WHEN 1 THEN 'MBRs intersect'
    END AS MBR_INTERSECTS
  FROM sample_polys a, sample_polys b
  WHERE a.id <= b.id
```

Resultados:

```
ID         ID          MBR_INTERSECTS
---------- --------- --------------
         1         1 MBRs intersect
         1         2 MBRs intersect
         2         2 MBRs intersect
         1         3 MBRs do not intersect
         2         3 MBRs do not intersect
         3         3 MBRs intersect
```

Exemplo 2:

A instrução SELECT a seguir determina se os retângulos de limite mínimo para as geometrias fazem interseção com o polígono com ID = 2.

```
SELECT a.id, b.id, ST_MBRIntersects (a.geometry, b.geometry) MBR_INTERSECTS
  FROM sample_polys a, sample_polys b
  WHERE a.id = 2
```

Resultados

```
ID         ID          MBR_INTERSECTS
---------- ----------- ---------------
         2           1               1
         2           2               1
         2           3               0
```

Referência Relacionada:

- "ST_EnvIntersects" na página 383
- "ST_MBR" na página 433

ST_MeasureBetween, ST_LocateBetween

ST_MeasureBetween ou ST_LocateBetween utiliza uma geometria e duas coordenadas M (medidas) como parâmetros de entrada e retorna essa parte da geometria especificada que representa o conjunto de caminhos ou pontos desconectados entre as duas coordenadas M.

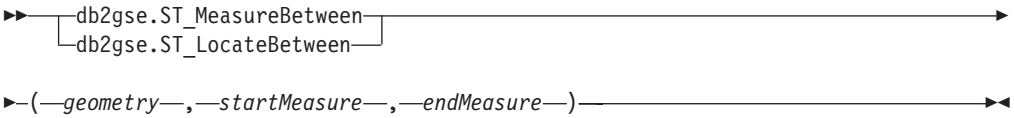
Para curvas, multicurvas, superfícies e multisuperfícies, a interpolação é executada para calcular o resultado. A geometria resultante é representada no sistema de referência espacial da geometria especificada.

Se a geometria especificada for uma superfície ou multisuperfície, ST_MeasureBetween ou ST_LocateBetween será aplicada aos anéis externo e interno da geometria. Se nenhuma das partes da geometria especificada estiver no intervalo definido pelas coordenadas M especificadas, será retornada uma geometria vazia. Se a geometria especificada for nula, então nulo é retornado.

A geometria resultante é representada no tipo espacial mais apropriado. Se puder ser representada como um ponto, cadeia de linhas ou polígono, será utilizado um desses tipos. Caso contrário, será utilizado o tipo multiponto, cadeia multilinha ou multipolígono.

As duas funções também podem ser chamadas como métodos.

Sintaxe:



Parâmetros:

geometry
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria na qual as partes com valores de medidas entre *startMeasure* e *endMeasure* devem ser encontradas.

startMeasure
: Um valor do tipo DOUBLE que representa o limite inferior do intervalo de medidas. Se este valor for nulo, não será aplicado nenhum limite inferior.

endMeasure
: Um valor do tipo DOUBLE que representa o limite superior do intervalo de medidas. Se este valor for nulo, não será aplicado nenhum limite superior.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

A coordenada M (medida) de uma geometria é definida pelo usuário. Ela é muito versátil porque pode representar qualquer coisa que você deseja medir; por exemplo, a distância em uma rodovia, temperatura, pressão ou medidas de pH.

Este exemplo ilustra a utilização da coordenada M para registrar dados coletados de medidas de pH. Um pesquisador coleta o pH do solo em uma rodovia em locais específicos. Seguindo seus procedimentos operacionais padrão, ele anota os

valores necessários em cada local do qual ele retira uma amostra do solo: as coordenadas X e Y desse local e o pH medido por ele.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, geometry ST_LineString)

INSERT INTO sample_lines
  VALUES (1, ST_LineString ('linestring m (2 2 3, 3 5 3,
                                           3 3 6, 4 4 6,
                                           5 5 6, 6 6 8)', 1 ) )
```

Para encontrar o local em que a acidez do solo varia entre 4 e 6, o pesquisador utiliza esta instrução SELECT:

```
SELECT id, CAST( ST_AsText( ST_MeasureBetween( 4, 6) )
  AS VARCHAR(150) ) MEAS_BETWEEN_4_AND_6
  FROM sample_lines
```

Resultados:

```
ID         MEAS_BETWEEN_4_AND_6
---------- ------------------------------------------------------
         1 LINESTRING M (3.00000000 4.33333300 4.00000000,
                         3.00000000 3.00000000 6.00000000,
                         4.00000000 4.00000000 6.00000000,
                         5.00000000 5.00000000 6.00000000)
```

ST_MidPoint

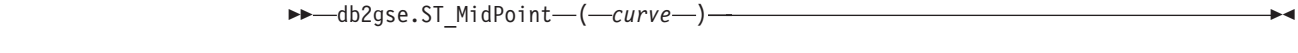
ST_MidPoint utiliza uma curva como um parâmetro de entrada e retorna o ponto na curva que é eqüidistante de ambos os pontos de extremidade da curva, medido ao longo da curva. O ponto resultante é representado no sistema de referência espacial da curva especificada.

Se a curva especificada for vazia, então um ponto vazio será retornado. Se a curva especificada for nula, será retornado nulo.

Se a curva contiver coordenadas Z ou coordenadas M (medidas), o ponto médio será determinado somente pelos valores das coordenadas X e Y na curva. A coordenada Z e a medida no ponto retornado são interpoladas.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

curve Um valor do tipo ST_Curve ou um de seus subtipos que representa a curva para a qual o ponto no meio é retornado.

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

Este exemplo ilustra a utilização de ST_MidPoint para retornar o ponto médio de curvas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, geometry ST_LineString)

INSERT INTO sample_lines (id, geometry)
  VALUES (1, ST_LineString ('linestring (0 0, 0 10, 0 20, 0 30, 0 40)', 1 ) )

INSERT INTO sample_lines (id, geometry)
  VALUES (2, ST_LineString ('linestring (2 2, 3 5, 3 3, 4 4, 5 5, 6 6)', 1 ) )

INSERT INTO sample_lines (id, geometry)
  VALUES (3, ST_LineString ('linestring (0 10, 0 0, 10 0, 10 10)', 1 ) )

INSERT INTO sample_lines (id, geometry)
  VALUES (4, ST_LineString ('linestring (0 20, 5 20, 10 20, 15 20)', 1 ) )

SELECT id, CAST( ST_AsText( ST_MidPoint(geometry) ) AS VARCHAR(60) ) MID_POINT
  FROM sample_lines
```

Resultados:

```
ID         MID_POINT
---------- ------------------------------------
         1 POINT ( 0.00000000 20.00000000)
         2 POINT ( 3.00000000 3.45981800)
         3 POINT ( 5.00000000 0.00000000)
         4 POINT ( 7.50000000 20.00000000)
```

ST_MinM

ST_MinM utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada M mínima.

Se a geometria especificada for nula ou vazia, ou se não tiver coordenadas M, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos para o qual a coordenada M mínima é retornada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_MinM. Três polígonos são criados e inseridos na tabela SAMPLE_POLYS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon zm ((110 120 20 3,
                                      110 140 22 3,
```

```
                                      120 130 26 4,
                                      110 120 20 3))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon zm ((0 0 40 7,
                                      0 4 35 9,
                                      5 4 32 12,
                                      5 0 31 5,
                                      0 0 40 7))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon('polygon zm ((12 13 10 16,
                                      8 4 10 12,
                                      9 4 12 11,
                                      12 13 10 16))', 0) )
```

Exemplo 1:

Este exemplo encontra a coordena M mínima de cada polígono em SAMPLE_POLYS.

```
SELECT id, CAST ( ST_MinM(geometry) AS INTEGER) MIN_M
  FROM sample_polys
```

Resultados:

```
ID          MIN_M
----------- ------------
          1            3
          2            5
          3           11
```

Exemplo 2:

Este exemplo encontra a coordenada M mínima existente para todos os polígonos na coluna GEOMETRY.

```
SELECT CAST ( MIN ( ST_MinM(geometry) ) AS INTEGER) OVERALL_MIN_M
  FROM sample_polys
```

Resultados:

```
OVERALL_MIN_M
-------------
            3
```

Referência Relacionada:

- “ST_MaxM” na página 426
- “ST_MinX” na página 439
- “ST_MinY” na página 441
- “ST_MinZ” na página 442

ST_MinX

ST_MinX utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada X mínima.

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos para o qual a coordenada X mínima é retornada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_MinX. Três polígonos são criados e inseridos na tabela SAMPLE_POLYS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon zm ((110 120 20 3,
                                      110 140 22 3,
                                      120 130 26 4,
                                      110 120 20 3))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon zm ((0 0 40 7,
                                      0 4 35 9,
                                      5 4 32 12,
                                      5 0 31 5,
                                      0 0 40 7))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon('polygon zm ((12 13 10 16,
                                      8 4 10 12,
                                      9 4 12 11,
                                      12 13 10 16))', 0) )
```

Exemplo 1:

Este exemplo encontra a coordena X mínima de cada polígono em SAMPLE_POLYS.

```
SELECT id, CAST ( ST_MinX(geometry) AS INTEGER) MIN_X
  FROM sample_polys
```

Resultados:

```
ID          MIN_X
----------- ------------
          1          110
          2            0
          3            8
```

Exemplo 2:

Este exemplo encontra a coordenada X mínima existente para todos os polígonos na coluna GEOMETRY.

```
SELECT CAST ( MIN ( ST_MinX(geometry) ) AS INTEGER) OVERALL_MIN_X
  FROM sample_polys
```

Resultados:

```
OVERALL_MIN_X
-------------
            0
```

Conceitos Relacionados:

- "ST_MaxX" na página 428

Referência Relacionada:

- "ST_MinM" na página 438
- "ST_MinY" na página 441
- "ST_MinZ" na página 442

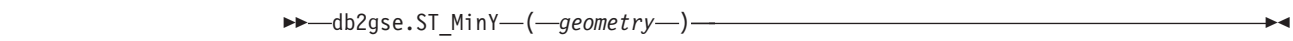
ST_MinY

ST_MinY utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada Y mínima.

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos para o qual a coordenada Y mínima é retornada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_MinY. Três polígonos são criados e inseridos na tabela SAMPLE_POLYS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon zm ((110 120 20 3,
                                      110 140 22 3,
                                      120 130 26 4,
                                      110 120 20 3))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon zm ((0 0 40 7,
                                      0 4 35 9,
                                      5 4 32 12,
                                      5 0 31 5,
                                      0 0 40 7))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon('polygon zm ((12 13 10 16,
```

```
                                        8 4 10 12,
                                        9 4 12 11,
                                        12 13 10 16))', 0) )
```

Exemplo 1:

Este exemplo encontra a coordena Y mínima de cada polígono em SAMPLE_POLYS.

```
SELECT id, CAST ( ST_MinY(geometry) AS INTEGER) MIN_Y
  FROM sample_polys
```

Resultados:

```
ID          MIN_Y
----------- ------------
          1          120
          2            0
          3            4
```

Exemplo 2:

Este exemplo encontra a coordenada Y mínima existente para todos os polígonos na coluna GEOMETRY.

```
SELECT CAST ( MIN ( ST_MinY(geometry) ) AS INTEGER) OVERALL_MIN_Y
  FROM sample_polys
```

Resultados:

```
OVERALL_MIN_Y
--------------
            0
```

Referência Relacionada:

- "ST_MaxY" na página 430
- "ST_MinM" na página 438
- "ST_MinX" na página 439
- "ST_MinZ" na página 442

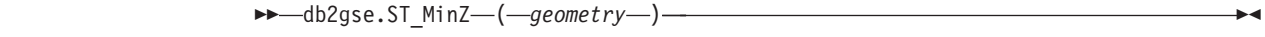
ST_MinZ

ST_MinZ utiliza uma geometria como um parâmetro de entrada e retorna sua coordenada Z mínima.

Se a geometria especificada for nula ou vazia, ou se não tiver coordenadas Z, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos para o qual a coordenada Z mínima é retornada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_MinZ. Três polígonos são criados e inseridos na tabela SAMPLE_POLYS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon zm ((110 120 20 3,
                                      110 140 22 3,
                                      120 130 26 4,
                                      110 120 20 3))', 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon zm ((0 0 40 7,
                                      0 4 35 9,
                                      5 4 32 12,
                                      5 0 31 5,
                                      0 0 40 7))', 0) )

INSERT INTO sample_polys
  VALUES (3, ST_Polygon('polygon zm ((12 13 10 16,
                                      8 4 10 12,
                                      9 4 12 11,
                                      12 13 10 16))', 0) )
```

Exemplo 1:

Este exemplo encontra a coordena Z mínima de cada polígono em SAMPLE_POLYS.

```
SELECT id, CAST ( ST_MinZ(geometry) AS INTEGER) MIN_Z
  FROM sample_polys
```

Resultados:

```
ID          MIN_Z
----------- ------------
          1           20
          2           31
          3           10
```

Exemplo 2:

Este exemplo encontra a coordenada Z mínima existente para todos os polígonos na coluna GEOMETRY.

```
SELECT CAST ( MIN ( ST_MinZ(geometry) ) AS INTEGER) OVERALL_MIN_Z
  FROM sample_polys
```

Resultados:

```
OVERALL_MIN_Z
-------------
           10
```

Referência Relacionada:

- "ST_MaxZ" na página 431
- "ST_MinM" na página 438

- "ST_MinX" na página 439
- "ST_MinY" na página 441

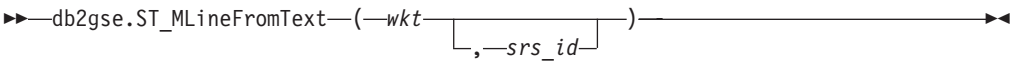
ST_MLineFromText

ST_MLineFromText utiliza uma representação de texto reconhecida de uma cadeia multilinha e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna a cadeia multilinha correspondente.

Se a representação de texto reconhecida for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_MultiLineString. Ela é recomendada por sua flexibilidade: ST_MultiLineString utiliza formatos adicionais de entrada, além da representação de texto reconhecida.

Sintaxe:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida da cadeia multilinha resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial para a cadeia multilinha resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se o *srs_id* especificado não identificar um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então uma condição de exceção é criada (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_MultiLineString

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MLineFromText pode ser utilizado para criar e inserir uma cadeia multilinha a partir de sua representação de texto reconhecida. O registro inserido tem ID = 1110 e a geometria é uma cadeia multilinha no sistema de referência espacial 1. A cadeia multilinha está na representação de texto reconhecida de uma cadeia multilinha. As coordenadas X e Y desta geometria são:

- Linha 1: (33, 2) (34, 3) (35, 6)
- Linha 2: (28, 4) (29, 5) (31, 8) (43, 12)
- Linha 3: (39, 3) (37, 4) (36, 7)

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mlines (id INTEGER, geometry ST_MultiLineString)

INSERT INTO sample_mlines
  VALUES (1110, ST_MLineFromText ('multilinestring ( (33 2, 34 3, 35 6),
                                                     (28 4, 29 5, 31 8, 43 12),
                                                     (39 3, 37 4, 36 7) )', 1) )
```

A instrução SELECT a seguir retorna a cadeia multilinha que foi registrada na tabela:

```
SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(280) ) MULTI_LINE_STRING
  FROM sample_mlines
  WHERE id = 1110
```

Resultados:

```
ID         MULTI_LINE_STRING
---------- ------------------------------------------------------------------
      1110 MULTILINESTRING (( 33.00000000 2.00000000, 34.00000000 3.00000000,
             35.00000000 6.00000000),
                           ( 28.00000000 4.00000000, 29.00000000 5.00000000,
             31.00000000 8.0000000, 43.00000000 12.00000000),
                           ( 39.00000000 3.00000000, 37.00000000 4.00000000,
             36.00000000 7.00000000 ))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_MLineFromWKB" na página 445
- "ST_MultiLineString" na página 453

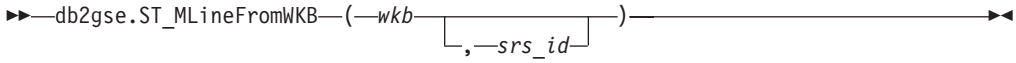
ST_MLineFromWKB

ST_MLineFromWKB utiliza uma representação binária reconhecida de uma cadeia multilinha e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna a cadeia multilinha correspondente.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_MultiLineString. Ela é recomendada por sua flexibilidade: ST_MultiLineString utiliza formatos adicionais de entrada, além da representação binária reconhecida.

Sintaxe:



Parâmetros:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida da cadeia multilinha resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial para a cadeia multilinha resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se o *srs_id* especificado não identificar um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então uma condição de exceção é criada (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_MultiLineString

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MLineFromWKB pode ser utilizado para criar uma cadeia multilinha a partir de sua representação binária reconhecida. A geometria é uma cadeia multilinha no sistema de referência espacial 1. Neste exemplo, a cadeia multilinha é armazenada com ID = 10 na coluna GEOMETRY da tabela SAMPLE_MLINES e, em seguida, a coluna WKB é atualizada com sua representação binária reconhecida (utilizando a função ST_AsBinary). Por último, a função ST_MLineFromWKB é utilizada para retornar a cadeia multilinha da coluna WKB. As coordenadas X e Y desta geometria são:

- Linha 1: (61, 2) (64, 3) (65, 6)
- Linha 2: (58, 4) (59, 5) (61, 8)
- Linha 3: (69, 3) (67, 4) (66, 7) (68, 9)

A tabela SAMPLE_MLINES tem uma coluna GEOMETRY, em que a cadeia multilinha é armazenada e uma coluna WKB, em que a representação binária reconhecida da cadeia multilinha é armazenada.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mlines (id INTEGER, geometry ST_MultiLineString,
       wkb BLOB(32K))

INSERT INTO sample_mlines
  VALUES (10, ST_MultiLineString ('multilinestring
       ( (61 2, 64 3, 65 6),
         (58 4, 59 5, 61 8),
         (69 3, 67 4, 66 7, 68 9) )', 1) )

UPDATE sample_mlines AS temporary_correlated
  SET wkb = ST_AsBinary( geometry )
  WHERE id = temporary_correlated.id
```

Na instrução SELECT a seguir, a função ST_MLineFromWKB é utilizada para recuperar a cadeia multilinha da coluna WKB.

```
SELECT id, CAST( ST_AsText( ST_MLineFromWKB (wkb) )
       AS VARCHAR(280) ) MULTI_LINE_STRING
  FROM sample_mlines
  WHERE id = 10
```

Resultados:

```
ID         MULTI_LINE_STRING
---------- ------------------------------------------------------------------
        10 MULTILINESTRING (( 61.00000000 2.00000000, 64.00000000 3.00000000,
              65.00000000 6.00000000),
                           ( 58.00000000 4.00000000, 59.00000000 5.00000000,
```

```
61.00000000 8.0000000),
                ( 69.00000000 3.00000000, 67.00000000 4.00000000,
66.00000000 7.00000000, 68.00000000 9.00000000 ))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_MLineFromText" na página 444
- "ST_MultiLineString" na página 453
- "Representação WKB (Well-Known Binary)" na página 524

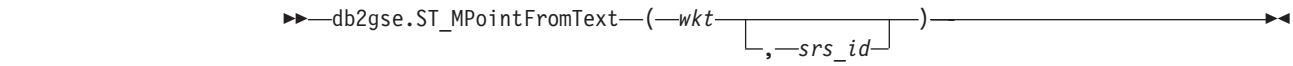
ST_MPointFromText

ST_MPointFromText utiliza uma representação de texto reconhecida de um multiponto e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna o multiponto correspondente.

Se a representação de texto reconhecida for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_MultiPoint. Ela é recomendada por sua flexibilidade: ST_MultiPoint utiliza formatos adicionais de entrada, além da representação de texto reconhecida.

Sintaxe:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do multiponto resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do multiponto resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se o *srs_id* especificado não identificar um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então uma condição de exceção é criada (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_MultiPoint

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MPointFromText pode ser utilizado para criar e inserir um multiponto a partir de sua representação de texto reconhecida. O

registro inserido tem ID = 1110 e a geometria é um multiponto no sistema de referência espacial 1. O multiponto está na representação de texto reconhecida de um multiponto. As coordenadas X e Y para esta geometria são: (1, 2) (4, 3) (5, 6).

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mpoints (id INTEGER, geometry ST_MultiPoint)

INSERT INTO sample_mpoints
  VALUES (1110, ST_MPointFromText ('multipoint (1 2, 4 3, 5 6) )', 1) )
```

A instrução SELECT a seguir retorna o multiponto que foi registrado na tabela:

```
SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(280) ) MULTIPOINT
  FROM sample_mpoints
  WHERE id = 1110
```

Resultados:

```
ID         MULTIPOINT
---------- ----------------------------------------------------------------
      1110 MULTIPOINT (1.00000000 2.00000000, 4.00000000 3.00000000,
                       5.00000000 6.00000000)
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_MPointFromWKB" na página 448
- "ST_MultiPoint" na página 455
- "Representação WKT (Well-Known Text)" na página 519

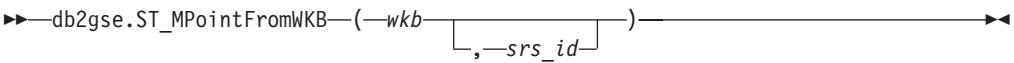
ST_MPointFromWKB

ST_MPointFromWKB utiliza uma representação binária reconhecida de um multiponto e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna o multiponto correspondente.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_MultiPoint. Ela é recomendada por sua flexibilidade: ST_MultiPoint utiliza formatos adicionais de entrada, além da representação binária reconhecida.

Sintaxe:



Parâmetros:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do multiponto resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do multiponto resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se o *srs_id* especificado não identificar um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então uma condição de exceção é criada (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_MultiPoint

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MPointFromWKB pode ser utilizado para criar um multiponto a partir de sua representação binária reconhecida. A geometria é um multiponto no sistema de referência espacial 1. Neste exemplo, o multiponto é armazenado com ID = 10 na coluna GEOMETRY da tabela SAMPLE_MPOINTS e, em seguida, a coluna WKB é atualizada com sua representação binária reconhecida (utilizando a função ST_AsBinary). Por último, a função ST_MPointFromWKB é utilizada para retornar o multiponto da coluna WKB. As coordenadas X e Y para esta geometria são: (44, 14) (35, 16) (24, 13).

A tabela SAMPLE_MPOINTS tem uma coluna GEOMETRY, em que o multiponto é armazenado e uma coluna WKB, em que a representação binária reconhecida do multiponto é armazenada.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mpoints (id INTEGER, geometry ST_MultiPoint,
    wkb BLOB(32K))

INSERT INTO sample_mpoints
  VALUES (10, ST_MultiPoint ('multipoint ( 4 14, 35 16, 24 13)', 1))

UPDATE sample_mpoints AS temporary_correlated
  SET wkb = ST_AsBinary( geometry )
  WHERE id = temporary_correlated.id
```

Na instrução SELECT a seguir, a função ST_MPointFromWKB é utilizada para recuperar o multiponto da coluna WKB.

```
SELECT id, CAST( ST_AsText( ST_MLineFromWKB (wkb)) AS VARCHAR(100)) MULTIPOINT
  FROM sample_mpoints
  WHERE id = 10
```

Resultados:

```
ID         MULTIPOINT
---------- ----------------------------------------------------------------
        10 MULTIPOINT (44.00000000 14.00000000, 35.00000000
              16.00000000 24.00000000 13.00000000)
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_MPointFromText" na página 447
- "ST_MultiPoint" na página 455

- "Representação WKB (Well-Known Binary)" na página 524
- "ST_Point" na página 468

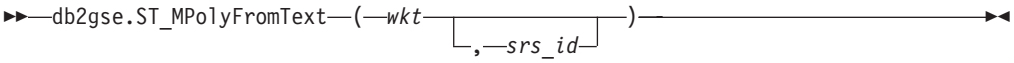
ST_MPolyFromText

ST_MPolyFromText utiliza uma representação de texto reconhecida de um multipolígono e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna o multipolígono correspondente.

Se a representação de texto reconhecida for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_MultiPolygon. Ela é recomendada por sua flexibilidade: ST_MultiPolygon utiliza formatos adicionais de entrada, além da representação de texto reconhecida.

Sintaxe:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do multipolígono resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do multipolígono resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se o *srs_id* especificado não identificar um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então uma condição de exceção é criada (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_MultiPolygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MPolyFromText pode ser utilizado para criar e inserir um multipolígono a partir de sua representação de texto reconhecida. O registro inserido tem ID = 1110 e a geometria é um multipolígono no sistema de referência espacial 1. O multipolígono está na representação de texto reconhecida de um multipolígono. As coordenadas X e Y desta geometria são:

- Polígono 1: (3, 3) (4, 6) (5, 3) (3, 3)
- Polígono 2: (8, 24) (9, 25) (1, 28) (8, 24)
- Polígono 3: (13, 33) (7, 36) (1, 40) (10, 43) (13, 33)

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mpolys (id INTEGER, geometry ST_MultiPolygon)

INSERT INTO sample_mpolys
  VALUES (1110,
          ST_MPolyFromText ('multipolygon (( (3 3, 4 6, 5 3, 3 3),
                                (8 24, 9 25, 1 28, 8 24),
                                (13 33, 7 36, 1 40, 10 43 13 33) ))', 1) )
```

A instrução SELECT a seguir retorna o multipolígono que foi registrado na tabela:

```
SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(350) ) MULTI_POLYGON
  FROM sample_mpolys
  WHERE id = 1110
```

Resultados:

```
ID         MULTI_POLYGON
------- ------------------------------------------------------------------------
   1110 MULTIPOLYGON ((( 13.00000000 33.00000000, 10.00000000 43.00000000,
           1.00000000 40.00000000, 7.00000000 36.00000000,
           13.00000000 33.00000000)),
                     (( 8.00000000 24.00000000, 9.00000000 25.00000000,
           1.00000000 28.0000000, 8.00000000 24.00000000)),
                      ( 3.00000000 3.00000000, 5.00000000 3.00000000,
           4.00000000 6.00000000, 3.00000000 3.00000000)))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_MPolyFromWKB" na página 451
- "ST_MultiPolygon" na página 456
- "Representação WKT (Well-Known Text)" na página 519

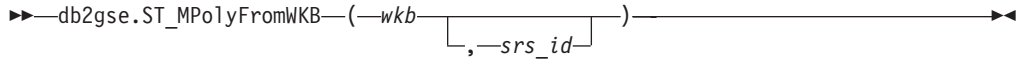
ST_MPolyFromWKB

ST_MPolyFromWKB utiliza uma representação binária reconhecida de um multipolígono e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna o multipolígono correspondente.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_MultiPolygon. Ela é recomendada por sua flexibilidade: ST_MultiPolygon utiliza formatos adicionais de entrada, além da representação binária reconhecida.

Sintaxe:



Parâmetros:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do multipolígono resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do multipolígono resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se o *srs_id* especificado não identificar um sistema de referência espacial listado na exibição de catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, então uma condição de exceção é criada (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_MultiPolygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MPolyFromWKB pode ser utilizado para criar um multipolígono a partir de sua representação binária reconhecida. A geometria é um multipolígono no sistema de referência espacial 1. Neste exemplo, o multipolígono é armazenado com ID = 10 na coluna GEOMETRY da tabela SAMPLE_MPOLYS e, em seguida, a coluna WKB é atualizada com sua representação binária reconhecida (utilizando a função ST_AsBinary). Por último, a função ST_MPolyFromWKB é utilizada para retornar o multipolígono da coluna WKB. As coordenadas X e Y desta geometria são:

- Polígono 1: (1, 72) (4, 79) (5, 76) (1, 72)
- Polígono 2: (10, 20) (10, 40) (30, 41) (10, 20)
- Polígono 3: (9, 43) (7, 44) (6, 47) (9, 43)

A tabela SAMPLE_MPOLYS tem uma coluna GEOMETRY, em que o multipolígono é armazenado e uma coluna WKB, em que a representação binária reconhecida do multipolígono é armazenada.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mpolys (id INTEGER,
    geometry ST_MultiPolygon, wkb BLOB(32K))

INSERT INTO sample_mpolys
  VALUES (10, ST_MultiPolygon ('multipolygon
        (( (1 72, 4 79, 5 76, 1 72),
         (10 20, 10 40, 30 41, 10 20),
         (9 43, 7 44, 6 47, 9 43) ))', 1))

UPDATE sample_mpolys AS temporary_correlated
  SET wkb = ST_AsBinary( geometry )
  WHERE id = temporary_correlated.id
```

Na instrução SELECT a seguir, a função ST_MPolyFromWKB é utilizada para recuperar o multipolígono da coluna WKB.

```
SELECT id, CAST( ST_AsText( ST_MPolyFromWKB (wkb) )
    AS VARCHAR(320) ) MULTIPOLYGON
  FROM sample_mpolys
  WHERE id = 10
```

Resultados:

```
ID         MULTIPOLYGON
---------- ------------------------------------------------------------------
        10 MULTIPOLYGON ((( 10.00000000 20.00000000, 30.00000000
             41.00000000, 10.00000000 40.00000000, 10.00000000
```

```
20.00000000)),
            ( 1.00000000 72.00000000, 5.00000000
76.00000000, 4.00000000 79.0000000, 1.00000000
72,00000000)),
             ( 9.00000000 43.00000000, 6.00000000
 47.00000000, 7.00000000 44.00000000, 9.00000000
 43.00000000 )))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_MPolyFromText" na página 450
- "ST_MultiPolygon" na página 456
- "Representação WKB (Well-Known Binary)" na página 524
- "ST_Polygon" na página 478

ST_MultiLineString

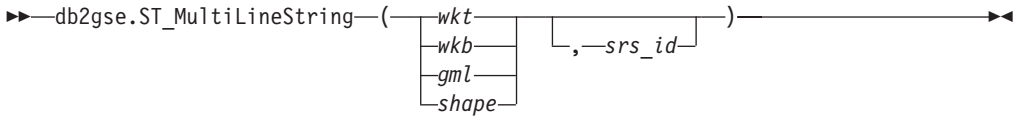
ST_MultiLineString constrói uma cadeia multilinha a partir de uma das seguintes entradas:

- Uma representação de texto reconhecida
- Uma representação binária reconhecida
- Uma representação de formatos
- Uma representação na GML (Geography Markup Language)

Um identificador do sistema de referência espacial opcional pode ser especificado para identificar o sistema de referência espacial no qual a cadeia multilinha resultante está localizada.

Se a representação de texto reconhecida, a representação binária reconhecida, a representação de formatos ou a representação GML for nula, será retornado nulo.

Sintaxe:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida da cadeia multilinha resultante.

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida da cadeia multilinha resultante.

gml Um valor do tipo CLOB(2G) que representa a cadeia multilinha resultante utilizando a linguagem de marcação geográfica.

shape Um valor do tipo BLOB(2G) que especifica a representação de formatos da cadeia multilinha resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial para a cadeia multilinha resultante.

Se o parâmetro *srs_id* for omitido, será utilizado o sistema de referência espacial com o identificador numérico 0 (zero).

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_MultiLineString

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MultiLineString pode ser utilizado para criar e inserir uma cadeia multilinha a partir de sua representação de texto reconhecida. O registro inserido tem ID = 1110 e a geometria é uma cadeia multilinha no sistema de referência espacial 1. A cadeia multilinha está na representação de texto reconhecida de uma cadeia multilinha. As coordenadas X e Y desta geometria são:

- Linha 1: (33, 2) (34, 3) (35, 6)
- Linha 2: (28, 4) (29, 5) (31, 8) (43, 12)
- Linha 3: (39, 3) (37, 4) (36, 7)

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mlines (id INTEGER,
                            geometry ST_MultiLineString)

INSERT INTO sample_mlines
  VALUES (1110,
          ST_MultiLineString ('multilinestring ( (33 2, 34 3, 35 6),
                                      (28 4, 29 5, 31 8, 43 12),
                                      (39 3, 37 4, 36 7) )', 1) )
```

A instrução SELECT a seguir retorna a cadeia multilinha que foi registrada na tabela:

```
SELECT id,
       CAST( ST_AsText( geometry ) AS VARCHAR(280) )
  MULTI_LINE_STRING
  FROM sample_mlines
  WHERE id = 1110
```

Resultados:

```
ID         MULTI_LINE_STRING
------- ------------------------------------------------------------------
   1110 MULTILINESTRING (( 33.00000000 2.00000000, 34.00000000 3.00000000,
              35.00000000 6.00000000),
                        ( 28.00000000 4.00000000, 29.00000000 5.00000000,
              31.00000000 8.0000000, 43.00000000 12.00000000),
                        ( 39.00000000 3.00000000, 37.00000000 4.00000000,
              36.00000000 7.00000000 ))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "Representação WKT (Well-Known Text)" na página 519
- "Representação WKB (Well-Known Binary)" na página 524
- "Representação de Formatos" na página 526

- "Representação GML (Geography Markup Language)" na página 526

ST_MultiPoint

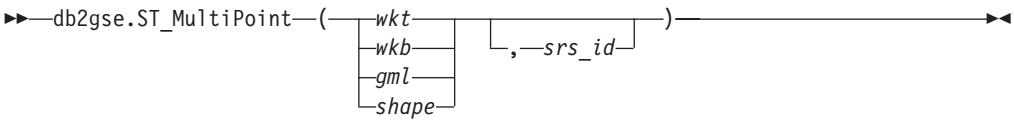
ST_MultiPoint constrói um multiponto a partir de uma das seguintes entradas:

- Uma representação de texto reconhecida
- Uma representação binária reconhecida
- Uma representação de formatos
- Uma representação na GML (Geography Markup Language)

Um identificador do sistema de referência espacial opcional pode ser especificado para indicar o sistema de referência espacial no qual o multiponto resultante está localizado.

Se a representação de texto reconhecida, a representação binária reconhecida, a representação de formatos ou a representação GML for nula, será retornado nulo.

Sintaxe:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do multiponto resultante.

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do multiponto resultante.

gml Um valor do tipo CLOB(2G) que representa o multiponto resultante utilizando a linguagem de marcação geográfica.

shape Um valor do tipo BLOB(2G) que especifica a representação de formatos do multiponto resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do multiponto resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MultiPoint pode ser utilizado para criar e inserir um multiponto a partir de sua representação de texto reconhecida. O registro inserido tem ID = 1110 e a geometria é um multiponto no sistema de referência espacial 1. O multiponto está na representação de texto reconhecida de um multiponto. As coordenadas X e Y para esta geometria são: (1, 2) (4, 3) (5, 6).

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mpoints (id INTEGER, geometry ST_MultiPoint)

INSERT INTO sample_mpoints
  VALUES (1110, ST_MultiPoint ('multipoint (1 2, 4 3, 5 6) )', 1))
```

A instrução SELECT a seguir retorna o multiponto que foi registrado na tabela:

```
SELECT id, CAST( ST_AsText(geometry) AS VARCHAR(90)) MULTIPOINT
  FROM sample_mpoints
  WHERE id = 1110
```

Resultados:

```
ID         MULTIPOINT
---------- --------------------------------------------------
      1110 MULTIPOINT (1.00000000 2.00000000, 4.00000000
           3.00000000, 5.00000000 6.00000000)
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "Representação WKT (Well-Known Text)" na página 519
- "Representação WKB (Well-Known Binary)" na página 524
- "Representação de Formatos" na página 526
- "Representação GML (Geography Markup Language)" na página 526

ST_MultiPolygon

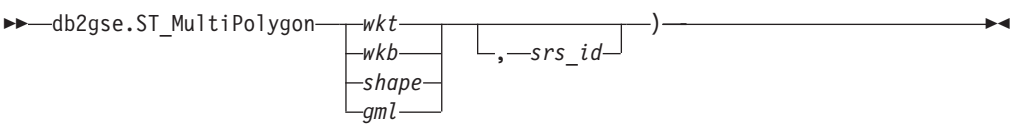
ST_MultiPolygon constrói um multipolígono a partir de uma das seguintes entradas:

- Uma representação de texto reconhecida
- Uma representação binária reconhecida
- Uma representação de formatos
- Uma representação na GML (Geography Markup Language)

Um identificador do sistema de referência espacial opcional pode ser especificado para identificar o sistema de referência espacial no qual o multipolígono resultante está localizado.

Se a representação de texto reconhecida, a representação binária reconhecida, a representação de formatos ou a representação GML for nula, será retornado nulo.

Sintaxe:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do multipolígono resultante.

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do multipolígono resultante.

gml Um valor do tipo CLOB(2G) que representa o multipolígono resultante utilizando a linguagem de marcação geográfica.

shape Um valor do tipo BLOB(2G) que representa a representação de formatos do multipolígono resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do multipolígono resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_MultiPolygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_MultiPolygon pode ser utilizado para criar e inserir um multipolígono a partir de sua representação de texto reconhecida. O registro inserido tem ID = 1110 e a geometria é um multipolígono no sistema de referência espacial 1. O multipolígono está na representação de texto reconhecida de um multipolígono. As coordenadas X e Y desta geometria são:

- Polígono 1: (3, 3) (4, 6) (5, 3) (3, 3)
- Polígono 2: (8, 24) (9, 25) (1, 28) (8, 24)
- Polígono 3: (13, 33) (7, 36) (1, 40) (10, 43) (13, 33)

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mpolys (id INTEGER, geometry ST_MultiPolygon)

INSERT INTO sample_mpolys
  VALUES (1110,
          ST_MultiPolygon ('multipolygon (( (3 3, 4 6, 5 3, 3 3),
                                    (8 24, 9 25, 1 28, 8 24),
                                    (13 33, 7 36, 1 40, 10 43 13 33) ))', 1) )
```

A instrução SELECT a seguir retorna o multipolígono que foi registrado na tabela:

```
SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(350) ) MULTI_POLYGON
  FROM sample_mpolys
  WHERE id = 1110
```

Resultados:

```
ID          MULTI_POLYGON
------- ------------------------------------------------------------------
   1110 MULTIPOLYGON ((( 13.00000000 33.00000000, 10.00000000 43.00000000,
           1.00000000 40.00000000, 7.00000000 36.00000000,
          13.00000000 33.00000000)),
```

```
          (( 8.00000000 24.00000000, 9.00000000 25.00000000,
1.00000000 28.0000000, 8.00000000 24.00000000)),
          (( 3.00000000 3.00000000, 5.00000000 3.00000000,
4.00000000 6.00000000, 3.00000000 3.00000000)))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "Representação WKT (Well-Known Text)" na página 519
- "Representação WKB (Well-Known Binary)" na página 524
- "Representação de Formatos" na página 526
- "Representação GML (Geography Markup Language)" na página 526

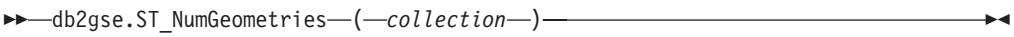
ST_NumGeometries

ST_NumGeometries utiliza uma coleção de geometrias como parâmetro de entrada e retorna o número de geometrias na coleção.

Se a coleção de geometrias especificada for nula ou vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

collection
Um valor do tipo ST_GeomCollection ou um de seus subtipos que representa a coleção de geometrias para a qual o número de geometrias é retornado.

Tipo de Retorno:

INTEGER

Exemplo:

Duas coleções de geometrias são armazenadas na tabela SAMPLE_GEOMCOLL. Uma é um multipolígono e a outra é um multiponto. A função ST_NumGeometries determina quantas geometrias individuais estão em cada coleção de geometrias.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geomcoll (id INTEGER, geometry ST_GeomCollection)

INSERT INTO sample_geomcoll
VALUES( 1,
         ST_MultiPolygon ('multipolygon (( (3 3, 4 6, 5 3, 3 3),
                               (8 24, 9 25, 1 28, 8 24),
                               (13 33, 7 36, 1 40, 10 43, 13 33) ))', 1) )

INSERT INTO sample_geomcoll
  VALUES (2, ST_MultiPoint ('multipoint (1 2, 4 3, 5 6, 7 6, 8 8)', 1) )

SELECT id, ST_NumGeometries (geometry) NUM_GEOMS_IN_COLL
  FROM sample_geomcoll
```

Resultados:

```
ID          NUM_GEOMS_IN_COLL
----------- -----------------
          1                 3
          2                 5
```

Referência Relacionada:

- "ST_GeometryN" na página 399

ST_NumInteriorRing

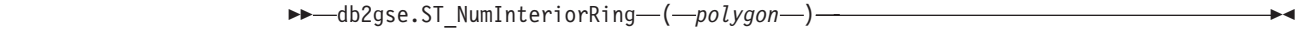
ST_NumInteriorRing utiliza um polígono como parâmetro de entrada e retorna o número de seus anéis internos.

Se o polígono especificado for nulo ou vazio, será retornado nulo.

Se o polígono não tiver anéis internos, será retornado 0 (zero).

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

polygon
: Um valor do tipo ST_Polygon que representa o polígono para o qual o número de anéis internos é retornado.

Tipo de Retorno:

INTEGER

Exemplo:

O exemplo a seguir cria dois polígonos:

- Um com dois anéis internos
- Um sem nenhum anel interno

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon('polygon
        ((40 120, 90 120, 90 150, 40 150, 40 120),
         (50 130, 60 130, 60 140, 50 140, 50 130),
         (70 130, 80 130, 80 140, 70 140, 70 130))' , 0) )

INSERT INTO sample_polys
  VALUES (2, ST_Polygon('polygon ((5 15, 50 15, 50 105, 5 15))' , 0) )
```

A função ST_NumInteriorRing é utilizada para retornar o número de anéis nas geometrias na tabela:

```
SELECT id, ST_NumInteriorRing(geometry) NUM_RINGS
FROM sample_polys
```

Resultados:

```
ID         NUM_RINGS
---------- --------------
         1              2
         2              0
```

Referência Relacionada:

- "ST_InteriorRingN" na página 406

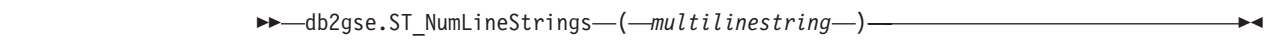
ST_NumLineStrings

ST_NumLineStrings utiliza uma cadeia multilinha como parâmetro de entrada e retorna o número de suas cadeias de linhas.

Se a cadeia multilinha especificada for nula ou vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

multilinestring
Um valor do tipo ST_MultiLineString que representa a cadeia de linhas múltiplas para a qual o número de cadeias de linhas é retornado.

Tipo de Retorno:

INTEGER

Exemplo:

As cadeias multilinhas são armazenadas na tabela SAMPLE_MLINES. A função ST_NumLineStrings determina quantas geometrias individuais estão em cada cadeia multilinha.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mlines (id INTEGER, geometry ST_MultiLineString)

INSERT INTO sample_mlines
  VALUES (110, ST_MultiLineString ('multilinestring
       ( (33 2, 34 3, 35 6),
         (28 4, 29 5, 31 8, 43 12),
         (39 3, 37 4, 36 7))', 1) )
INSERT INTO sample_mlines
  VALUES (111, ST_MultiLineString ('multilinestring
       ( (3 2, 4 3, 5 6),
         (8 4, 9 5, 3 8, 4 12))', 1) )

SELECT id, ST_NumLineStrings (geometry) NUM_WITHIN
  FROM sample_mlines
```

Resultados:

```
ID          NUM_WITHIN
----------- ----------
        110          3
        111          2
```

Referência Relacionada:

- "ST_LineStringN" na página 424

ST_NumPoints

ST_NumPoints utiliza uma geometria como parâmetro de entrada e retorna o número de pontos que foram utilizados para definir essa geometria. Por exemplo, se a geometria for um polígono e se foram utilizados cinco pontos para definir esse polígono, o número retornado será 5.

Se a geometria especificada for nula ou vazia, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria para a qual o número de pontos é retornado.

Tipo de Retorno:

INTEGER

Exemplo:

Várias geometrias são armazenadas na tabela. A função ST_NumPoints determina quantos pontos estão em cada geometria na tabela SAMPLE_GEOMETRIES.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (spatial_type VARCHAR(18), geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES ('st_point',
           ST_Point (2, 3, 0) )

INSERT INTO sample_geometries
  VALUES ('st_linestring',
           ST_LineString ('linestring (2 5, 21 3, 23 10)', 0) )

INSERT INTO sample_geometries
  VALUES ('st_polygon',
           ST_Polygon ('polygon ((110 120, 110 140, 120 130, 110 120))', 0) )

SELECT spatial_type, ST_NumPoints (geometry) NUM_POINTS
  FROM sample_geometries
```

Resultados:

```
SPATIAL_TYPE    NUM_POINTS
--------------- ----------
st_point                 1
st_linestring            3
st_polygon               4
```

Referência Relacionada:

- "ST_PointN" na página 474

ST_NumPolygons

ST_NumPolygons utiliza um multipolígono como parâmetro de entrada e retorna o número de seus polígonos.

Se o multipolígono especificado for nulo ou vazio, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

multipolygon
Um valor do tipo ST_MultiPolygon que representa o multipolígono para o qual o número de polígonos é retornado.

Tipo de Retorno:

INTEGER

Exemplo:

Os multipolígonos são armazenados na tabela SAMPLE_MPOLYS. A função ST_NumPolygons determina quantas geometrias individuais estão em cada multipolígono.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mpolys (id INTEGER, geometry ST_MultiPolygon)

INSERT INTO sample_mpolys
  VALUES (1,
          ST_MultiPolygon ('multipolygon (( (3 3, 4 6, 5 3, 3 3),
                                (8 24, 9 25, 1 28, 8 24),
                                (13 33, 7 36, 1 40, 10 43, 13 33) ))', 1) )

INSERT INTO sample_polys
  VALUES (2,
          ST_MultiPolygon ('multipolygon empty', 1) )

INSERT INTO sample_polys
  VALUES (3,
          ST_MultiPolygon ('multipolygon (( (3 3, 4 6, 5 3, 3 3),
                                (13 33, 7 36, 1 40, 10 43, 13 33) ))', 1) )

SELECT id, ST_NumPolygons (geometry) NUM_WITHIN
  FROM sample_mpolys
```

Resultados:

```
ID          NUM_WITHIN
----------- ----------
          1          3
          2          0
          3          2
```

Referência Relacionada:

- "ST_PolygonN" na página 481

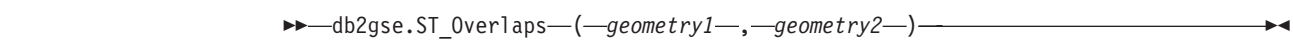
ST_Overlaps

ST_Overlaps utiliza duas geometrias como parâmetros de entrada e retorna 1 se a interseção das geometrias resultar em uma geometria com a mesma dimensão mas diferente de qualquer uma das geometrias especificadas. Caso contrário, será retornado 0 (zero).

Se qualquer uma das duas geometrias for nula ou vazia, será retornado nulo.

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Sintaxe:



Parâmetros:

geometry1
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada para sobreposição com *geometry2*.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada para sobreposição com *geometry1*.

Tipo de Retorno:

INTEGER

Exemplos:

Estes exemplos ilustram a utilização de ST_Overlaps. Várias geometrias são criadas e inseridas na tabela SAMPLE_GEOMETRIES

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES (1, ST_Point (10, 20, 1)),
         (2, ST_Point ('point (41 41)', 1) ),
         (10, ST_LineString ('linestring (1 10, 3 12, 10 10)', 1) ),
         (20, ST_LineString ('linestring (50 10, 50 12, 45 10)', 1) ),
         (30, ST_LineString ('linestring (50 12, 50 10, 60 8)', 1) ),
         (100, ST_Polygon ('polygon ((0 0, 0 40, 40 40, 40 0, 0 0))', 1) ),
         (110, ST_Polygon ('polygon ((30 10, 30 30, 50 30, 50 10, 30 10))', 1) ),
         (120, ST_Polygon ('polygon ((0 50, 0 60, 40 60, 40 60, 0 50))', 1) )
```

Exemplo 1:

Este exemplo encontra os IDs de pontos de sobreposição.

```
SELECT sg1.id, sg2.id
  CASE ST_Overlaps (sg1.geometry, sg2.geometry)
    WHEN 0 THEN 'Points_do_not_overlap'
    WHEN 1 THEN 'Points_overlap'
    END
```

```
  AS OVERLAP
  FROM sample_geometries sg1, sample_geometries sg2
  WHERE sg1.id < 10 AND sg2.id < 10 AND sg1.id >= sg2.id
```

Resultados:

```
ID          ID         OVERLAP
----------- ---------- ------------------------
          1          1 Points_do_not_overlap
          2          1 Points_do_not_overlap
          2          2 Points_do_not_overlap
```

Exemplo 2:

Este exemplo encontra os IDs de linhas de sobreposição.

```
SELECT sg1.id, sg2.id
  CASE ST_Overlaps (sg1.geometry, sg2.geometry)
    WHEN 0 THEN 'Lines_do_not_overlap'
    WHEN 1 THEN 'Lines_overlap'
    END
  AS OVERLAP
  FROM sample_geometries sg1, sample_geometries sg2
  WHERE sg1.id >= 10 AND sg1.id < 100
    AND sg2.id >= 10 AND sg2.id < 100
    AND sg1.id >= sg2.id
```

Resultados:

```
ID          ID         OVERLAP
----------- ---------- ------------------------
         10         10 Lines_do_not_overlap
         20         10 Lines_do_not_overlap
         30         10 Lines_do_not_overlap
         20         20 Lines_do_not_overlap
         30         20 Lines_overlap
         30         30 Lines_do_not_overlap
```

Exemplo 3:

Este exemplo encontra os IDs de polígonos de sobreposição.

```
SELECT sg1.id, sg2.id
  CASE ST_Overlaps (sg1.geometry, sg2.geometry)
    WHEN 0 THEN 'Polygons_do_not_overlap'
    WHEN 1 THEN 'Polygons_overlap'
    END
  AS OVERLAP
  FROM sample_geometries sg1, sample_geometries sg2
  WHERE sg1.id >= 100 AND sg2.id >= 100 AND sg1.id >= sg2.id
```

Resultados:

```
ID          ID         OVERLAP
----------- ---------- ------------------------
        100        100 Polygons_do_not_overlap
        110        100 Polygons_overlap
        120        100 Polygons_do_not_overlap
        110        110 Polygons_do_not_overlap
        120        110 Polygons_do_not_overlap
        120        120 Polygons_do_not_overlap
```

Referência Relacionada:

- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

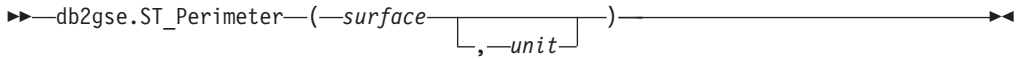
ST_Perimeter

ST_Perimeter utiliza uma superfície ou multisuperfície e, opcionalmente, uma unidade como parâmetros de entrada e retorna o perímetro da superfície ou multisuperfície, que é o comprimento de seu limite, medido nas unidades padrão ou especificadas.

Se a superfície ou multisuperfície especificada for nula ou vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

surface Um valor do tipo ST_Surface, ST_MultiSurface ou um de seus subtipos para o qual o perímetro é retornado.

unit Um valor VARCHAR(128) que identifica as unidades nas quais o perímetro é medido. As unidades de medida suportadas estão listadas na exibição do catálogo DB2GSE.ST_UNITS_OF_MEASURE.

Se o parâmetro *unit* for omitido, as seguintes regras serão utilizadas para determinar a unidade na qual o perímetro é medido:

- Se *surface* estiver em um sistema de coordenadas projetadas ou geocêntricas, a unidade linear associada a este sistema de coordenadas será a padrão.
- Se *surface* estiver em um sistema de coordenadas geográficas, mas não estiver em um SRS (Spatial Reference System) geodésico, a unidade angular associada a este sistema de coordenadas será a padrão.
- Se *surface* estiver em um SRS geodésico, a unidade de medida padrão será metros.

Restrições para conversões de unidades: Será retornado um erro (SQLSTATE 38SU4) se ocorrer qualquer uma das condições a seguir:

- A geometria está em um sistema de coordenadas não especificado e o parâmetro *unit* é especificado.
- A geometria está em um sistema de coordenadas projetadas e uma unidade angular é especificada.
- A geometria está em um sistema de coordenadas geográficas, mas não está em um SRS geodésico e uma unidade linear é especificada.
- A geometria está em um sistema de coordenadas geográficas, está em um SRS geodésico e uma unidade angular é especificada.

Tipo de Retorno:

DOUBLE

Exemplos:

Estes exemplos ilustram a utilização da função ST_Perimeter. É criado um sistema de referência espacial com um ID 4000 utilizando uma chamada para db2se, e é criado um polígono nesse sistema de referência espacial.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse

db2se create_srs se_bank -srsId 4000 -srsName new_york1983
  -xOffset 0 -yOffset 0 -xScale 1 -yScale 1
  -coordsysName NAD_1983_StatePlane_New_York_East_FIPS_3101_Feet
```

A tabela SAMPLE_POLYS é criada para conter uma geometria com um perímetro 18.

```
CREATE TABLE sample_polys (id SMALLINT, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1, ST_Polygon ('polygon ((0 0, 0 4, 5 4, 5 0, 0 0))', 4000))
```

Exemplo 1:

Este exemplo lista o ID e o perímetro do polígono.

```
SELECT id, ST_Perimeter (geometry) AS PERIMETER
FROM sample_polys
```

Resultados:

```
ID          PERIMETER
----------- ------------------------
          1 +1.80000000000000E+001
```

Exemplo 2:

Este exemplo lista o ID e o perímetro do polígono com o perímetro medido em metros.

```
SELECT id, ST_Perimeter (geometry, 'METER') AS PERIMETER_METER
FROM sample_polys
```

Resultados:

```
ID          PERIMETER_METER
----------- ------------------------
          1 +5.48641097282195E+000
```

ST_PerpPoints

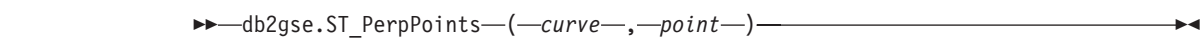
ST_PerpPoints utiliza uma curva ou multicurva e um ponto como parâmetros de entrada e retorna uma projeção perpendicular do ponto especificado na curva ou multicurva. É retornado o ponto com a menor distância entre o ponto especificado e o ponto perpendicular. Se dois ou mais desses pontos perpendiculares projetados forem eqüidistantes do ponto especificado, todos eles serão retornados. Se nenhum ponto perpendicular puder ser construído, será retornado um ponto vazio.

Se a curva ou multicurva especificada tiver coordenadas Z ou M, as coordenadas Z ou M dos pontos resultantes serão calculadas por interpolação na curva ou multicurva especificada.

Se a curva ou ponto especificado for vazio, será retornado um ponto vazio. Se a curva ou ponto for nulo, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

curve Um valor do tipo ST_Curve, ST_MultiCurve ou um de seus subtipos que representa a curva ou multicurva na qual a projeção perpendicular do *ponto* é retornada.

point Um valor do tipo ST_Point que representa o ponto perpendicular projetado em *curve*.

Tipo de Retorno:

db2gse.ST_MultiPoint

Exemplos:

Estes exemplos ilustram a utilização da função ST_PerpPoints para localizar pontos que são perpendiculares à cadeia de linhas armazenada na seguinte tabela. A função ST_LineString é utilizada na instrução INSERT para criar a cadeia de linhas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, line ST_LineString)

INSERT INTO sample_lines (id, line)
  VALUES (1, ST_LineString('linestring (0 10, 0 0, 10 0, 10 10)' , 0) )
```

Exemplo 1:

Este exemplo localiza a projeção perpendicular na cadeia de linhas de um ponto com coordenadas (5, 0). A função ST_AsText é utilizada para converter o valor retornado (um multiponto) em sua representação de texto reconhecida.

```
SELECT CAST ( ST_AsText( ST_PerpPoints( line, ST_Point(5, 0) ) )
  AS VARCHAR(50) ) PERP
  FROM sample_lines
```

Resultados:

```
PERP
----------------------------------------
MULTIPOINT ( 5.00000000 0.00000000)
```

Exemplo 2:

Este exemplo localiza as projeções perpendiculares na cadeia de linhas de um ponto com coordenadas (5, 5). Neste caso, existem três pontos na cadeia de linhas que estão eqüidistantes da localização especificada. Portanto, é retornado um multiponto que consiste nos três pontos.

```
SELECT  CAST ( ST_AsText( ST_PerpPoints( line, ST_Point(5, 5) ) )
  AS VARCHAR160) ) PERP
  FROM sample_lines
```

Resultados:

```
PERP
--------------------------------------------------------------------------------
MULTIPOINT ( 0.00000000 5.00000000, 5.00000000 0.00000000, 10.00000000 5.00000000)
```

Exemplo 3:

Este exemplo localiza as projeções perpendiculares na cadeia de linhas de um ponto com coordenadas (5, 10). Neste caso, existem três diferentes pontos perpendiculares que podem ser encontrados. No entanto, a função ST_PerpPoints somente retorna os pontos que estão mais próximos do ponto especificado. Assim, é retornado um multiponto que consiste apenas nos dois pontos mais próximos. O terceiro ponto não é incluído.

```
SELECT CAST ( ST_AsText( ST_PerpPoints( line, ST_Point(5, 10) ) )
  AS VARCHAR(80) ) PERP
  FROM sample_lines
```

Resultados:

```
PERP
------------------------------------------------------------
MULTIPOINT ( 0.00000000 10.00000000, 10.00000000 10.00000000 )
```

Exemplo 4:

Este exemplo localiza a projeção perpendicular na cadeia de linhas de um ponto com coordenadas (5, 15).

```
SELECT CAST ( ST_AsText( ST_PerpPoints( line, ST_Point('point(5 15)', 0 ) ) )
  AS VARCHAR(80) ) PERP
  FROM sample_lines
```

Resultados:

```
PERP
-----------------------------------
MULTIPOINT ( 5.00000000 0.00000000 )
```

Exemplo 5:

Neste exemplo, o ponto especificado com coordenadas (15 15) não possui nenhuma projeção perpendicular na cadeia de linhas. Portanto, é retornada uma geometria vazia.

```
SELECT CAST ( ST_AsText( ST_PerpPoints( line, ST_Point(15, 15) ) )
  AS VARCHAR(80) ) PERP
  FROM sample_lines
```

Resultados:

```
PERP
-----------------------------------------------
MULTIPOINT EMPTY
```

ST_Point

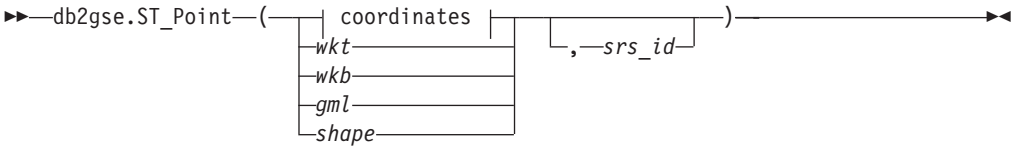
ST_Point constrói um ponto a partir de um dos seguintes conjuntos de entrada:

- Somente coordenadas X e Y
- Coordenadas X, Y e Z
- Coordenadas X, Y, Z e M
- Uma representação de texto reconhecida
- Uma representação binária reconhecida
- Uma representação de formatos
- Uma representação na GML (Geography Markup Language)

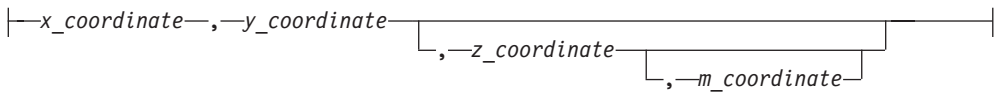
Um identificador do sistema de referência espacial opcional pode ser especificado para indicar o sistema de referência espacial no qual o ponto resultante está localizado.

Se o ponto for construído a partir de coordenadas e se a coordenada X ou Y for nula, ocorrerá uma condição de exceção (SQLSTATE 38SUP). Se a coordenada Z ou M for nula, o ponto resultante não terá uma coordenada Z ou M, respectivamente. Se o ponto for construído a partir de sua representação de texto reconhecida, de sua representação binária reconhecida, de sua representação de formatos ou de sua representação GML, e se a representação for nula, será retornado nulo.

Sintaxe:



coordinates:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do ponto resultante.

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do ponto resultante.

gml Um valor do tipo CLOB(2G) que representa o ponto resultante utilizando a linguagem de marcação geográfica.

shape Um valor do tipo BLOB(2G) que representa a representação de formatos do ponto resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do ponto resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

x_coordinate
Um valor do tipo DOUBLE que especifica a coordenada X para o ponto resultante.

y_coordinate
Um valor do tipo DOUBLE que especifica a coordenada Y para o ponto resultante.

z_coordinate
Um valor do tipo DOUBLE que especifica a coordenada Z para o ponto resultante.

Se o parâmetro *z_coordinate* for omitido, o ponto resultante não terá uma coordenada Z. O resultado de ST_Is3D será 0 (zero) para tal ponto.

m_coordinate
Um valor do tipo DOUBLE que especifica a coordenada M para o ponto resultante.

Se o parâmetro *m_coordinate* for omitido, o ponto resultante não terá uma medida. O resultado de ST_IsMeasured será 0 (zero) para tal ponto.

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Exemplo 1:

Este exemplo ilustra como ST_Point pode ser utilizado para criar e inserir pontos. O primeiro ponto é criado utilizando um conjunto de coordenadas X e Y. O segundo ponto é criado utilizando sua representação de texto reconhecida. Ambos os pontos são geometrias no sistema de referência espacial 1.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)

INSERT INTO sample_points
  VALUES (1100, ST_Point (10, 20, 1) )

INSERT INTO sample_points
  VALUES (1101, ST_Point ('point (30 40)', 1) )
```

A instrução SELECT a seguir retorna os pontos que foram registrados na tabela:

```
SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(90)) POINTS
 FROM sample_points
```

Resultados:

```
ID         POINTS
---------- -----------------------------------
      1110 POINT ( 10.00000000 20.00000000)
      1101 POINT ( 30.00000000 40.00000000)
```

Exemplo 2:

Este exemplo insere um registro na tabela SAMPLE_POINTS com o ID 1103 e um valor de ponto com uma coordenada X de 120, uma coordenada Y de 358, uma coordenada M de 34, mas nenhuma coordenada Z.

```
INSERT INTO SAMPLE_POINTS(ID, GEOMETRY)
 VALUES(1103, db2gse.ST_Point(120, 358, CAST(NULL AS DOUBLE), 34, 1))
 SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(90) ) POINTS
 FROM sample_points
```

Resultados:

```
ID         POINTS
 ---------- ----------------------------------------------
       1103 POINT M ( 120.0000000 358.0000000 34.00000000)
```

Exemplo 3:

Este exemplo insere uma linha na tabela SAMPLE_POINTS com ID 1104 e um valor de ponto com uma coordenada X de 1003, uma coordenada Y de 9876, uma coordenada Z de 20 e um sistema de referência espacial 0, utilizando a linguagem de marcação geográfica para sua representação.

```
INSERT INTO SAMPLE_POINTS(ID, GEOMETRY)
 VALUES(1104, db2gse.ST_Point('<gml:Point><gml:coord>
    <gml:x>1003</gml:X><gml:Y>9876</gml:Y><gml:Z>20</gml:Z>
    </gml:coord></gml:Point>', 1))
 SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(90) ) POINTS
 FROM sample_points
```

Resultados:

```
ID         POINTS
 ---------- ----------------------------------------------------
       1104 POINT Z ( 1003.000000 9876.000000 20.00000000)
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_Is3d" na página 410
- "ST_IsMeasured" na página 414
- "Representação WKT (Well-Known Text)" na página 519
- "Representação WKB (Well-Known Binary)" na página 524
- "Representação de Formatos" na página 526
- "Representação GML (Geography Markup Language)" na página 526

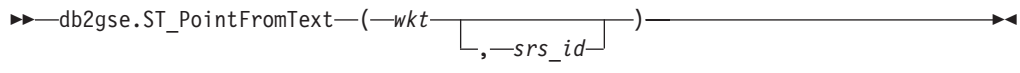
ST_PointFromText

ST_PointFromText utiliza uma representação de texto reconhecida de um ponto e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna o ponto correspondente.

Se a representação de texto reconhecida for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_Point. Ela é recomendada por sua flexibilidade: ST_Point utiliza formatos adicionais de entrada, além da representação de texto reconhecida.

Sintaxe:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do ponto resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do ponto resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

Este exemplo ilustra como ST_PointFromText pode ser utilizado para criar e inserir um ponto a partir de sua representação de texto reconhecida. O registro inserido tem ID = 1110 e a geometria é um ponto no sistema de referência espacial 1. O ponto está na representação de texto reconhecida de um ponto. As coordenadas X e Y para esta geometria são: (10, 20).

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)

INSERT INTO sample_points
  VALUES (1110, ST_PointFromText ('point (30 40)', 1) )
```

A instrução SELECT a seguir retorna o polígono que foi registrado na tabela:

```
SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(35) ) POINTS
  FROM sample_points
  WHERE id = 1110
```

Resultados:

```
ID         POINTS
---------- -------------------------------------
      1110 POINTS ( 30.00000000 40.00000000)
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_Point" na página 468
- "ST_PointFromWKB" na página 472

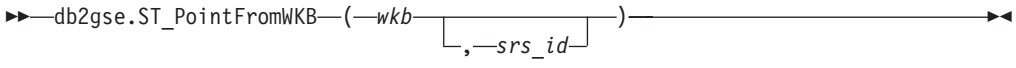
ST_PointFromWKB

ST_PointFromWKB utiliza uma representação binária reconhecida de um ponto e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna o ponto correspondente.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_Point. Ela é recomendada por sua flexibilidade: ST_Point utiliza formatos adicionais de entrada, além da representação binária reconhecida.

Sintaxe:



Parâmetros:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do ponto resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do ponto resultante.

Se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) será utilizado implicitamente.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

Este exemplo ilustra como ST_PointFromWKB pode ser utilizado para criar um ponto a partir de sua representação binária reconhecida. As geometrias são pontos no sistema de referência espacial 1. Neste exemplo, os pontos são armazenados na coluna GEOMETRY da tabela SAMPLE_POLYS e depois a coluna WKB é atualizada com suas representações binárias reconhecidas (utilizando a função ST_AsBinary). Por último, a função ST_PointFromWKB é utilizada para retornar os pontos da coluna WKB.

A tabela SAMPLE_POINTS tem uma coluna GEOMETRY, em que os pontos são armazenados e uma coluna WKB, em que as representações binárias reconhecidas dos pontos são armazenadas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point, wkb BLOB(32K))

INSERT INTO sample_points
  VALUES (10, ST_Point ('point (44 14)', 1) ),
  VALUES (11, ST_Point ('point (24 13)', 1))

UPDATE sample_points AS temporary_correlated
  SET wkb = ST_AsBinary( geometry )
  WHERE id = temporary_correlated.id
```

Na instrução SELECT a seguir, a função ST_PointFromWKB é utilizada para recuperar os pontos da coluna WKB.

```
SELECT id, CAST( ST_AsText( ST_PolyFromWKB (wkb) ) AS VARCHAR(35) ) POINTS
  FROM sample_points
```

Resultados:

```
ID         POINTS
---------- -----------------------------------
        10 POINT ( 44.00000000 14.00000000)
        11 POINT ( 24.00000000 13.00000000)
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_Point" na página 468

- "ST_PointFromText" na página 471

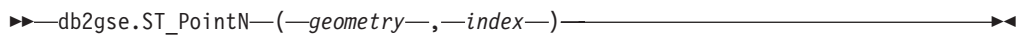
ST_PointN

ST_PointN utiliza uma cadeia de linhas ou um multiponto e um índice como parâmetros de entrada e retorna o ponto na cadeia de linhas ou multiponto que é identificado pelo índice. O ponto resultante é representado no sistema de referência espacial da cadeia de linhas ou multiponto especificado.

Se a cadeia de linhas ou o multiponto for nulo ou vazio, será retornado nulo. Se o índice for menor do que 1 ou maior do que o número de pontos na cadeia de linhas ou multiponto, será retornado nulo e também uma condição de aviso (SQLSTATE 01HS2).

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

geometry
Um valor do tipo ST_LineString ou ST_MultiPoint que representa a geometria a partir da qual o ponto identificado por *index* é retornado.

index Um valor do tipo INTEGER que identifica o *ú*ltimo ponto que será retornado de *geometry*.

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

O exemplo a seguir ilustra a utilização de ST_PointN.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, line ST_LineString)

INSERT INTO sample_lines
  VALUES (1, ST_LineString ('linestring (10 10, 5 5, 0 0, 10 0, 5 5, 0 10)', 0) )

SELECT id, CAST ( ST_AsText (ST_PointN (line, 2) ) AS VARCHAR(60) ) SECOND_INDEX
  FROM sample_lines
```

Resultados:

```
ID        SECOND_INDEX
--------- ------------------------------
        1 POINT (5.00000000 5.00000000)
```

Referência Relacionada:

- "ST_Endpoint" na página 381
- "ST_NumPoints" na página 461
- "ST_StartPoint" na página 486

ST_PointOnSurface

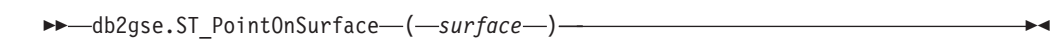
ST_PointOnSurface utiliza uma superfície ou multisuperfície como parâmetro de entrada e retorna um ponto que certamente estará no interior da superfície ou multisuperfície. Este ponto é o paracentróide da superfície.

O ponto resultante é representado no sistema de referência espacial da superfície ou multisuperfície especificada.

Se a superfície ou multisuperfície especificada for nula ou vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

surface Um valor do tipo ST_Surface, ST_MultiSurface ou um de seus subtipos que representa a geometria para a qual é retornado um ponto na superfície.

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

No exemplo a seguir, são criados dois polígonos e ST_PointOnSurface é utilizado. Um dos polígonos tem um orifício em seu centro. Os pontos retornados estão na superfície dos polígonos. Eles não estão necessariamente no centro dos polígonos.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1,
          ST_Polygon ('polygon ( (40 120, 90 120, 90 150, 40 150, 40 120) ,
                           (50 130, 80 130, 80 140, 50 140, 50 130) )' ,0) )
INSERT INTO sample_polys
  VALUES (2,
          ST_Polygon ('polygon ( (10 10, 50 10, 10 30, 10 10) )', 0) )

SELECT id, CAST (ST_AsText (ST_PointOnSurface (geometry) ) AS VARCHAR(80) )
  POINT_ON_SURFACE
  FROM sample_polys
```

Resultados:

```
ID          POINT_ON_SURFACE
----------- ----------------------------------
          1 POINT ( 65.00000000 125.00000000)
          2 POINT ( 30.00000000 15.00000000)
```

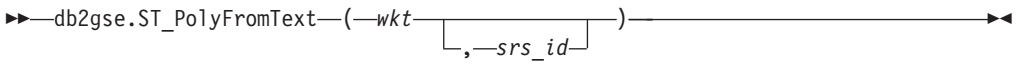
ST_PolyFromText

ST_PolyFromText utiliza uma representação de texto reconhecida de um polígono e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna o polígono correspondente.

Se a representação de texto reconhecida for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_Polygon. Ela é recomendada por sua flexibilidade: ST_Polygon utiliza formatos adicionais de entrada, além da representação de texto reconhecida.

Sintaxe:



Parâmetros:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do polígono resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do polígono resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Polygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_PolyFromText pode ser utilizado para criar e inserir um polígono a partir de sua representação de texto reconhecida. O registro inserido tem ID = 1110 e a geometria é um polígono no sistema de referência espacial 1. O polígono está na representação de texto reconhecida de um polígono. As coordenadas X e Y para esta geometria são: (50, 20) (50, 40) (70, 30).

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1110, ST_PolyFromText ('polygon ((50 20, 50 40, 70 30, 50 20))', 1) )
```

A instrução SELECT a seguir retorna o polígono que foi registrado na tabela:

```
SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(120) ) POLYGON
FROM sample_polys
  WHERE id = 1110
```

Resultados:

```
ID         POLYGON
---------- ------------------------------------------------------------------
      1110 POLYGON (( 50.00000000 20.00000000, 70.00000000 30.00000000,
                      50.00000000 40.00000000, 50.00000000 20.00000000))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_PolyFromWKB" na página 477
- "ST_Polygon" na página 478

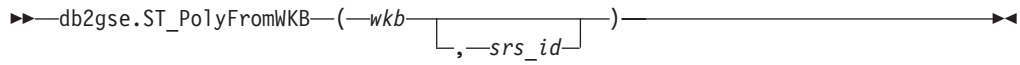
ST_PolyFromWKB

ST_PolyFromWKB utiliza uma representação binária reconhecida de um polígono e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada e retorna o polígono correspondente.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

A função recomendada para alcançar o mesmo resultado é a função ST_Polygon. Ela é recomendada por sua flexibilidade: ST_Polygon utiliza formatos adicionais de entrada, além da representação binária reconhecida.

Sintaxe:



Parâmetros:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do polígono resultante.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do polígono resultante.

Se o parâmetro *srs_id* for o omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Polygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_PolyFromWKB pode ser utilizado para criar um polígono a partir de sua representação binária reconhecida. A geometria é um

polígono no sistema de referência espacial 1. Neste exemplo, o polígono é armazenado com ID = 1115 na coluna GEOMETRY da tabela SAMPLE_POLYS e, em seguida, a coluna WKB é atualizada com sua representação binária reconhecida (utilizando a função ST_AsBinary). Por último, a função ST_PolyFromWKB é utilizada para retornar o multipolígono da coluna WKB. As coordenadas X e Y para esta geometria são: (50, 20) (50, 40) (70, 30).

A tabela SAMPLE_POLYS tem uma coluna GEOMETRY, em que o polígono é armazenado e uma coluna WKB, em que a representação binária reconhecida do polígono é armazenada.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon,
  wkb BLOB(32K))

INSERT INTO sample_polys
  VALUES (10, ST_Polygon ('polygon ((50 20, 50 40, 70 30, 50 20))', 1) )

UPDATE sample_polys AS temporary_correlated
  SET wkb = ST_AsBinary( geometry )
  WHERE id = temporary_correlated.id
```

Na instrução SELECT a seguir, a função ST_PolyFromWKB é utilizada para recuperar o polígono da coluna WKB.

```
SELECT id, CAST( ST_AsText( ST_PolyFromWKB (wkb) )
  AS VARCHAR(120) ) POLYGON
  FROM sample_polys
  WHERE id = 1115
```

Resultados:

```
ID         POLYGON
---------- ------------------------------------------------------------
      1115 POLYGON (( 50.00000000 20.00000000, 70.00000000
              30.00000000,50.00000000 40.00000000, 50.00000000
              20.00000000))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_PolyFromText" na página 476
- "ST_Polygon" na página 478

ST_Polygon

ST_Polygon constrói um polígono a partir de uma das seguintes entradas:

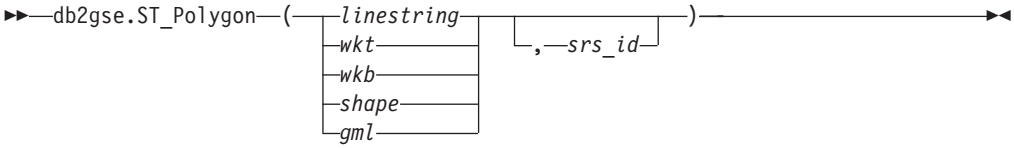
- Uma cadeia de linhas fechada que define o anel externo do polígono resultante
- Uma representação de texto reconhecida
- Uma representação binária reconhecida
- Uma representação de formatos
- Uma representação na GML (Geography Markup Language)

Um identificador do sistema de referência espacial opcional pode ser especificado para identificar o sistema de referência espacial no qual o polígono resultante está localizado.

Se o polígono for construído a partir de uma cadeia de linhas e a cadeia de linhas especificada for nula, será retornado nulo. Se a cadeia de linhas especificada estiver vazia, será retornado um polígono vazio. Se o polígono for construído a partir de sua representação de texto reconhecida, de sua representação binária reconhecida, de sua representação de formatos ou de sua representação GML, e se a representação for nula, será retornado nulo.

Esta função também pode ser chamada como um método somente nos seguintes casos: ST_Polygon(*linestring*) e ST_Polygon(*linestring*, *srs_id*).

Sintaxe:



Parâmetros:

linestring
: Um valor do tipo ST_LineString que representa a cadeia de linhas que define o anel externo do limite externo. Se *linestring* não estiver fechado e for simples, ocorrerá uma condição de exceção (SQLSTATE 38SSL).

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida do polígono resultante.

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida do polígono resultante.

shape Um valor do tipo BLOB(2G) que representa a representação de formatos do polígono resultante.

gml Um valor do tipo CLOB(2G) que representa o polígono resultante utilizando a linguagem de marcação geográfica.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial do polígono resultante.

Se o polígono for construído a partir de um parâmetro *linestring* especificado e o parâmetro *srs_id* for omitido, o sistema de referência espacial de *linestring* será utilizado implicitamente. Caso contrário, se o parâmetro *srs_id* for omitido, o sistema de referência espacial com o identificador numérico 0 (zero) é utilizado.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Polygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_Polygon pode ser utilizado para criar e inserir polígonos. Três polígonos são criados e inseridos. Todos eles são geometrias no sistema de referência espacial 1.

- O primeiro polígono é criado a partir de um anel (uma cadeia de linhas fechada e simples). As coordenadas X e Y para este polígono são: (10, 20) (10, 40) (20, 30).
- O segundo polígono é criado utilizando sua representação de texto reconhecida. As coordenadas X e Y para este polígono são: (110, 120) (110, 140) (120, 130).
- O terceiro polígono é um polígono circular. Um polígono circular consiste em um polígono interno e um externo. Este polígono circular é criado utilizando sua representação de texto reconhecida. As coordenadas X e Y para o polígono externo são: (110, 120) (110, 140) (130, 140) (130, 120) (110, 120). As coordenadas X e Y para o polígono interno são: (115, 125) (115, 135) (125, 135) (125, 135) (115, 125).

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
  VALUES (1100,
          ST_Polygon (ST_LineString ('linestring
                     (10 20, 10 40, 20 30, 10 20)',1), 1))

INSERT INTO sample_polys
  VALUES (1101,
          ST_Polygon ('polygon
                    ((110 120, 110 140, 120 130, 110 120))', 1))

INSERT INTO sample_polys
  VALUES (1102,
          ST_Polygon ('polygon
                    ((110 120, 110 140, 130 140, 130 120, 110 120),
                    (115 125, 115 135, 125 135, 125 135, 115 125))', 1))
```

A instrução SELECT a seguir retorna os polígonos que foram registrados na tabela:

```
SELECT id, CAST( ST_AsText( geometry ) AS VARCHAR(120) ) POLYGONS
FROM sample_polys
```

Resultados:

```
ID          POLYGONS
------- ------------------------------------------------------------
   1110 POLYGON (( 10.00000000 20.00000000, 20.00000000 30.00000000
                   10.00000000 40.00000000, 10.00000000 20.00000000))

   1101 POLYGON (( 110.00000000 120.00000000, 120.00000000 130.00000000
                   110.00000000 140.00000000, 110.00000000 120.00000000))

   1102 POLYGON (( 110.00000000 120.00000000, 130.00000000 120.00000000
                   130.00000000 140.00000000, 110.00000000 140.00000000
                   110.00000000 120.00000000),
                 ( 115.00000000 125.00000000, 115.00000000 135.00000000
                   125.00000000 135.00000000, 125.00000000 135.00000000
                   115.00000000 125.00000000))
```

Conceitos Relacionados:

- “Dados Espaciais e Geodésicos” na página 4

Referência Relacionada:

- “Representação WKT (Well-Known Text)” na página 519
- “Representação WKB (Well-Known Binary)” na página 524
- “Representação de Formatos” na página 526

- "Representação GML (Geography Markup Language)" na página 526

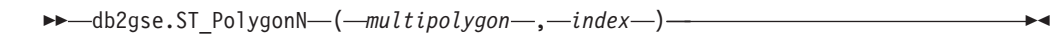
ST_PolygonN

ST_PolygonN utiliza um multipolígono e um índice como parâmetros de entrada e retorna o polígono que é identificado pelo índice. O polígono resultante é representado no sistema de referência espacial do multipolígono especificado.

Se o multipolígono especificado for nulo ou vazio, ou se o índice for menor do que 1 ou maior do que o número de polígonos, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

multipolygon
Um valor do tipo ST_MultiPolygon que representa o multipolígono a partir do qual o polígono identificado por *index* é retornado.

index Um valor do tipo INTEGER que identifica o *ú*ltimo polígono que será retornado de *multipolygon*.

Tipo de Retorno:

db2gse.ST_Polygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados variará, de acordo com sua exibição on-line

Este exemplo ilustra a utilização de ST_PolygonN.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_mpolys (id INTEGER, geometry ST_MultiPolygon)

INSERT INTO sample_mpolys
  VALUES (1, ST_Polygon ('multipolygon (((3 3, 4 6, 5 3, 3 3),
                                         (8 24, 9 25, 1 28, 8 24)
                                         (13 33, 7 36, 1 40, 10 43,
                                          13 33)))', 1))

SELECT id, CAST ( ST_AsText (ST_PolygonN (geometry, 2) )
    AS VARCHAR(120) ) SECOND_INDEX
FROM sample_mpolys
```

Resultados:

```
ID        SECOND_INDEX
--------- ------------------------------------------------------------
        1 POLYGON (( 8.00000000 24.00000000, 9.00000000 25.00000000,
                     1.00000000 28.00000000, 8.00000000 24.00000000))
```

Referência Relacionada:

- "ST_NumPolygons" na página 462

ST_Relate

ST_Relate utiliza duas geometrias e uma matriz DE-9IM como parâmetros de entrada e retorna 1 se as geometrias especificadas atenderem às condições especificadas pela matriz. Caso contrário, será retornado 0 (zero).

Se qualquer uma das geometrias especificadas for nula ou vazia, será retornado nulo.

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetros:

geometry1
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada em *geometry2*.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada contra *geometry1*.

matrix Um valor do tipo CHAR(9) que representa a matriz DE-9IM que será utilizada para o teste de *geometry1* e *geometry2*.

Tipo de Retorno:

INTEGER

Exemplo:

O código a seguir cria dois polígonos separados. Em seguida, a função ST_Relate é utilizada para determinar vários relacionamentos entre os dois polígonos. Por exemplo, se os dois polígonos são sobrepostos.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_polys (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_polys
VALUES( 1,
          ST_Polygon('polygon ( (40 120, 90 120, 90 150, 40 150, 40 120) )', 0))
INSERT INTO sample_polys
VALUES(2,
          ST_Polygon('polygon ( (30 110, 50 110, 50 130, 30 130, 30 110) )', 0))

SELECT ST_Relate(a.geometry, b.geometry, 'T*T***T**')  "Overlaps  ",
       ST_Relate(a.geometry, b.geometry, 'T*T***FF*')  "Contains  ",
       ST_Relate(a.geometry, b.geometry, 'T*F**F***')  "Within    "
       ST_Relate(a.geometry, b.geometry, 'T********')  "Intersects",
       ST_Relate(a.geometry, b.geometry, 'T*F**FFF2')  "Equals    "
  FROM sample_polys a, sample_polys b
  WHERE a.id = 1 AND b.id = 2
```

Resultados:

```
Overlaps    Contains    Within      Intersects  Equals
----------- ----------- ----------- ----------- -----------
          1           0           0           1           0
```

Referência Relacionada:

- "Funções que Comparam Recursos Geográficos" na página 308

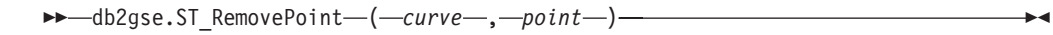
ST_RemovePoint

ST_RemovePoint utiliza uma curva e um ponto como parâmetros de entrada e retorna a curva especificada com todos os pontos iguais ao ponto especificado removido dela. Se a curva especificada tiver coordenadas Z ou M, o ponto também deverá ter coordenadas Z ou M. A geometria resultante é representada no sistema de referência espacial da geometria especificada.

Se a curva especificada for vazia, então um curva vazia será retornada. Se a curva especificada for nula, ou se o ponto especificado for nulo ou vazio, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

curve Um valor do tipo ST_Curve ou um de seus subtipos que representa a curva a partir da qual *point* é removido.

point Um valor do tipo ST_Point que identifica os pontos que são removidos de *curve*.

Tipo de Retorno:

db2gse.ST_Curve

Exemplos:

No exemplo a seguir, duas cadeias de linhas são incluídas na tabela SAMPLE_LINES. Estas cadeias de linhas são utilizadas nos exemplos abaixo.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, line ST_LineString)

INSERT INTO sample_lines
  VALUES (1,ST_LineString('linestring
         (10 10, 5 5, 0 0, 10 0, 5 5, 0 10)', 0))

INSERT INTO sample_lines
  VALUES (2, ST_LineString('linestring z
         (0 0 4, 5 5 5, 10 10 6, 5 5 7, 0 0 8)', 0))
```

Nos exemplos a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Exemplo 1:

O exemplo a seguir remove o ponto (5, 5) da cadeia de linhas que tem ID = 1. Este ponto ocorre duas vezes na cadeia de linhas. Portanto, as duas ocorrências são removidas.

```
SELECT CAST(ST_AsText (ST_RemovePoint (line, ST_Point(5, 5) ) )
  AS VARCHAR(120) ) RESULT
  FROM sample_lines
  WHERE id = 1
```

Resultados:

```
RESULT
------------------------------------------------------------
LINESTRING ( 10.00000000 10.00000000, 0.00000000 0.00000000,
             10.00000000 0.00000000, 0.00000000 10.00000000)
```

Exemplo 2:

O exemplo a seguir remove o ponto (5, 5, 5) da cadeia de linhas que tem ID = 2. Este ponto ocorre somente uma vez, portanto, somente essa ocorrência é removida.

```
SELECT CAST (ST_AsText (ST_RemovePoint (line, ST_Point(5.0, 5.0, 5.0)))
  AS VARCHAR(160) ) RESULT
  FROM sample_lines
  WHERE id=2
```

Resultados:

```
RESULT
------------------------------------------------------------------------
LINESTRING Z ( 0.00000000 0.00000000 4.00000000, 10.00000000 10.00000000
               6.00000000,  5.00000000 5.00000000 7.00000000, 0.00000000
               0.00000000 8.00000000)
```

ST_SrsId, ST_SRID

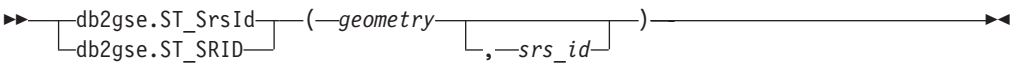
ST_SrsId (ou ST_SRID) utiliza uma geometria e, opcionalmente, um identificador do sistema de referência espacial como parâmetros de entrada. O que ela retorna depende de quais parâmetros de entrada foram especificados:

- Se o identificador do sistema de referência espacial for especificado, será retornada uma geometria com seu sistema de referência espacial alterado para o sistema de referência espacial especificado. Nenhuma transformação da geometria é executada.
- Se nenhum identificador do sistema de referência espacial for especificado como parâmetro de entrada, será retornado o identificador do sistema de referência espacial atual da geometria especificada.

Se a geometria especificada for nula, então nulo é retornado.

Estas funções também podem ser chamadas como métodos.

Sintaxe:



Parâmetros:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria para a qual o identificador do sistema de referência espacial será definido ou retornado.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial que será utilizado para a geometria resultante.

Atenção: Se este parâmetro for especificado, a geometria não será transformada, mas será retornada com seu sistema de referência espacial alterado para o sistema de referência espacial especificado. Como resultado da alteração para o novo sistema de referência espacial, os dados poderão estar danificados. Para transformações, utilize ST_Transform.

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipos de Retorno:

- INTEGER, se um srs_id não for especificado
- db2gse.ST_Geometry, se um srs_id for especificado

Exemplo:

São criados dois pontos em dois sistemas de referência espacial diferentes. O ID do sistema de referência espacial que está associado a cada ponto pode ser encontrado utilizando a função ST_SrsId.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)

INSERT INTO sample_points
  VALUES (1, ST_Point( 'point (80 180)', 0 ) )
INSERT INTO sample_points
  VALUES (2, ST_Point( 'point (-74.21450127 + 42.03415094)', 1 ) )

SELECT id, ST_SRSId (geometry) SRSID
  FROM sample_points
```

Resultados:

```
ID          SRSID
----------- -----------
          1           0
          2           1
```

Referência Relacionada:

- "ST_Transform" na página 498

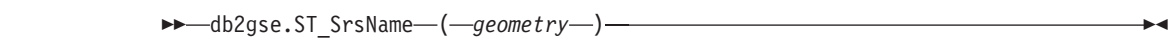
ST_SrsName

ST_SrsName utiliza uma geometria como um parâmetro de entrada e retorna o nome do sistema de referência espacial no qual a geometria especificada é representada.

Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria para a qual o nome do sistema de referência espacial é retornado.

Tipo de Retorno:

VARCHAR(128)

Exemplo:

São criados dois pontos em sistemas de referência espacial diferentes. A função ST_SrsName é utilizada para encontrar o nome do sistema de referência espacial que está associado a cada ponto.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry, ST_Point)

INSERT INTO sample_points
  VALUES (1, ST_Point ('point (80 180)', 0) )

INSERT INTO sample_points
  VALUES (2, ST_Point ('point (-74.21450127 + 42.03415094)', 1) )

SELECT id, ST_SrsName (geometry) SRSNAME
  FROM sample_points
```

Resultados:

```
ID          SRSNAME
----------- ------------------------------
          1 DEFAULT_SRS
          2 NAD83_SRS_1
```

Referência Relacionada:

- "ST_SrsId, ST_SRID" na página 484

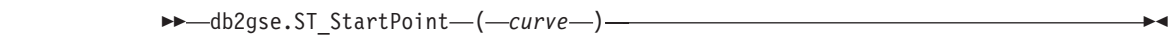
ST_StartPoint

ST_StartPoint utiliza uma curva como parâmetro de entrada e retorna o ponto que é o primeiro ponto da curva. O ponto resultante é representado no sistema de referência espacial da curva especificada. Este resultado é equivalente à chamada de função ST_PointN(*curve*, 1)

Se a curva especificada for nula ou vazia, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

curve Um valor do tipo ST_Curve ou um de seus subtipos que representa a geometria a partir da qual o primeiro ponto é retornado.

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

No exemplo a seguir, duas cadeias de linhas são incluídas na tabela SAMPLE_LINES. A primeira é uma cadeia de linhas com coordenadas X e Y. A segunda é uma cadeia de linhas com coordenadas X, Y e Z. A função ST_StartPoint é utilizada para retornar o primeiro ponto em cada cadeia de linhas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_lines (id INTEGER, line ST_LineString)

INSERT INTO sample_lines
  VALUES (1, ST_LineString ('linestring
         (10 10, 5 5, 0 0, 10 0, 5 5, 0 10)', 0))

INSERT INTO sample_lines
  VALUES (1, ST_LineString ('linestring z
         (0 0 4, 5 5 5, 10 10 6, 5 5 7, 0 0 8)', 0))

SELECT id, CAST( ST_AsText( ST_StartPoint( line ) ) AS VARCHAR(80))
  START_POINT
  FROM sample_lines
```

Resultados:

```
ID          START_POINT
----------- --------------------------------------------
          1 POINT ( 10.00000000 10.00000000)
          2 POINT Z ( 0.00000000 0.00000000 4.00000000)
```

Referência Relacionada:

- "ST_Endpoint" na página 381
- "ST_PointN" na página 474

ST_SymDifference

ST_SymDifference utiliza duas geometrias como parâmetros de entrada e retorna a geometria que é a diferença simétrica das duas geometrias. A diferença simétrica é a parte que não faz interseção das duas geometrias especificadas. A geometria resultante é representada no sistema de referência espacial da primeira geometria.A dimensão da geometria retornada é igual à das geometrias de entrada. As duas geometrias devem ter a mesma dimensão.

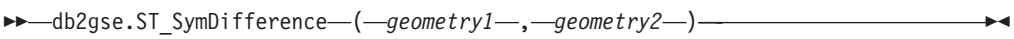
Para dados não geodésicos, se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial. Para dados geodésicos, as duas geometrias devem estar no mesmo sistema de referência espacial (SRS) geodésico.

Se as geometrias forem iguais, será retornada uma geometria vazia do tipo ST_Point.Se qualquer uma das geometrias for nula, será retornado nulo.

A geometria resultante é representada no tipo espacial mais apropriado. Se puder ser representada como um ponto, cadeia de linhas ou polígono, será utilizado um desses tipos. Caso contrário, será utilizado o tipo multiponto, cadeia multilinha ou multipolígono.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

geometry1
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a primeira geometria para calcular a diferença simétrica com *geometry2*.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a segunda geometria para calcular a diferença simétrica com *geometry1*.

Restrições para dados geodésicos:

- As duas geometrias devem ser geodésicas e devem estar no mesmo SRS geodésico.
- ST_SymDifference suporta apenas os tipos de dados ST_Point, ST_Polygon, ST_MultiPoint e ST_MultiPolygon.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplos:

Estes exemplos ilustram a utilização da função ST_SymDifference. As geometrias são armazenadas na tabela SAMPLE_GEOMS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geoms
VALUES( 1,
          ST_Geometry ('polygon ( (10 10, 10 20, 20 20, 20 10, 10 10) )', 0))

INSERT INTO sample_geoms
  VALUES
    (2, ST_Geometry ('polygon ( (30 30, 30 50, 50 50, 50 30, 30 30) )', 0))

INSERT INTO sample_geoms
  VALUES
    (3,ST_Geometry ('polygon ( (40 40, 40 60, 60 60, 60 40, 40 40) )', 0))

INSERT INTO sample_geoms
  VALUES
    (4, ST_Geometry ('linestring (70 70, 80 80)' , 0) )

INSERT INTO sample_geoms
  VALUES
    (5, ST_Geometry('linestring(75 75, 90 90)' ,0));
```

Nos exemplos a seguir, os resultados foram reformatados para possibilitar a leitura. Seus resultados variarão, de acordo com sua exibição.

Exemplo 1:

Este exemplo utiliza ST_SymDifference para retornar a diferença simétrica de dois polígonos separados na tabela SAMPLE_GEOMS.

```
SELECT a.id, b.id,
    CAST (ST_AsText (ST_SymDifference (a.geometry, b.geometry) )
    AS VARCHAR(350) ) SYM_DIFF
  FROM sample_geoms a, sample_geoms b
  WHERE a.id = 1 AND b.id = 2
```

Resultados:

```
ID   ID  SYM_DIFF
----- ----- ------------------------------------------------------------------
    1     2 MULTIPOLYGON ((( 10.00000000 10.00000000, 20.00000000 10.00000000,
                20.00000000 20.00000000, 10.00000000 20.00000000,
                10.00000000 10.00000000)),
                         (( 30.00000000 30.00000000, 50.00000000 30.00000000,
                50.00000000 50.00000000, 30.00000000 50.00000000,
              30.00000000 30.00000000)))
```

Exemplo 2:

Este exemplo utiliza ST_SymDifference para retornar a diferença simétrica de dois polígonos que fazem interseção na tabela SAMPLE_GEOMS.

```
SELECT a.id, b.id,
    CAST (ST_AsText (ST_SymDifference (a.geometry, b.geometry) )
    AS VARCHAR(500) ) SYM_DIFF
  FROM sample_geoms a, sample_geoms b
  WHERE a.id = 2 AND b.id = 3
```

Resultados:

```
ID   ID  SYM_DIFF
--- --- ------------------------------------------------------------------
  2   3  MULTIPOLYGON ((( 40.00000000 50.00000000, 50.00000000 50.00000000,
            50.00000000 40.00000000, 60.00000000 40.00000000,
            60.00000000 60.00000000, 40.00000000 60.00000000,
            40.00000000 50.00000000)),
                         (( 30.00000000 30.00000000, 50.00000000 30.00000000,
            50.00000000 40.00000000, 40.00000000 40.00000000,
            40.00000000 50.00000000, 30.00000000 50.00000000,
            30.00000000 30.00000000)))
```

Exemplo 3:

Este exemplo utiliza ST_SymDifference para retornar a diferença simétrica de duas cadeias de linhas que fazem interseção na tabela SAMPLE_GEOMS.

```
SELECT a.id, b.id,
    CAST (ST_AsText (ST_SymDifference (a.geometry, b.geometry) )
    AS VARCHAR(350) ) SYM_DIFF
  FROM sample_geoms a, sample_geoms b
  WHERE a.id = 4 AND b.id = 5
```

Resultados:

```
ID   ID  SYM_DIFF
---- --- ------------------------------------------------------------------
   4   5 MULTILINESTRING (( 70.00000000 70.00000000, 75.00000000 75.00000000),
               ( 80.00000000 80.00000000, 90.00000000 90.00000000))
```

Referência Relacionada:

- "ST_Difference" na página 369

ST_ToGeomColl

ST_ToGeomColl utiliza uma geometria como um parâmetro de entrada e a converte em uma coleção de geometria. A coleção de geometria resultante é representada no sistema de referência espacial da geometria especificada.

Se a geometria especificada for vazia, ela poderá ser de qualquer tipo. No entanto, ela será convertida em ST_Multipoint, ST_MultiLineString ou ST_MultiPolygon conforme apropriado.

Se a geometria especificada não for vazia, ela deverá ser do tipo ST_Point, ST_LineString ou ST_Polygon. Elas serão convertidas em ST_Multipoint, ST_MultiLineString ou ST_MultiPolygon, respectivamente.

Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que é convertida para uma coleção de geometria.

Tipo de Retorno:

db2gse.ST_GeomCollection

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados variará, de acordo com sua exibição on-line

Este exemplo ilustra a utilização da função ST_ToGeomColl.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES (1, ST_Polygon ('polygon ((3 3, 4 6, 5 3, 3 3))', 1)),
         (2, ST_Point ('point (1 2)', 1))
```

Na instrução SELECT a seguir, a função ST_ToGeomColl é utilizada para retornar geometrias como seus subtipos de coleção de geometria correspondentes.

```
SELECT id, CAST( ST_AsText( ST_ToGeomColl(geometry) )
    AS VARCHAR(120) ) GEOM_COLL
FROM   sample_geometries
```

Resultados:

```
ID          GEOM_COLL
----------- ----------------------------------------------------
          1 MULTIPOLYGON ((( 3.00000000 3.00000000, 5.00000000
```

```
                                3.00000000, 4.00000000 6.00000000,
                                3.00000000 3.00000000)))
          2 MULTIPOINT ( 1.00000000 2.00000000)
```

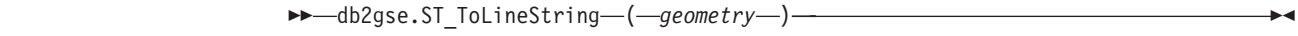
ST_ToLineString

ST_ToLineString utiliza uma geometria como um parâmetro de entrada e a converte em uma cadeia de linhas. A cadeia de linhas resultante é representada no sistema de referência espacial da geometria especificada.

A geometria especificada deve ser vazia ou uma cadeia de linhas. Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que é convertida para uma cadeia de linhas.

Uma geometria pode ser convertida em uma cadeia de linhas se for vazia ou uma cadeia de linhas. Se a conversão não puder ser executada, ocorrerá uma condição de exceção (SQLSTATE 38SUD).

Tipo de Retorno:

db2gse.ST_LineString

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados variará, de acordo com sua exibição on-line

Este exemplo ilustra a utilização da função ST_ToLineString.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES (1, ST_Geometry ('linestring z (0 10 1, 0  0 3, 10 0 5)', 0)),
         (2, ST_Geometry ('point empty', 1) ),
         (3, ST_Geometry ('multipolygon empty', 1) )
```

Na instrução SELECT a seguir, a função ST_ToLineString é utilizada para retornar cadeias de linhas convertidas em ST_LineString a partir do tipo estático de ST_Geometry.

```
SELECT CAST( ST_AsText( ST_ToLineString(geometry) )
     AS VARCHAR(130) ) LINES
FROM   sample_geometries
```

Resultados:

```
LINES
------------------------------------------------------------------
LINESTRING Z ( 0.00000000 10.00000000 1.00000000, 0.00000000
               0.00000000 3.00000000, 10.00000000 0.00000000
               5.00000000)
LINESTRING EMPTY
LINESTRING EMPTY
```

ST_ToMultiLine

ST_ToMultiLine utiliza uma geometria como um parâmetro de entrada e a converte em uma cadeia de múltiplas linhas. A cadeia de múltiplas linhas resultante é representada no sistema de referência espacial da geometria especificada.

A geometria especificada deve ser vazia, uma cadeia multilinha ou uma cadeia de linhas. Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que é convertida para uma cadeia de linhas múltiplas.

Uma geometria pode ser convertida em uma cadeia multilinha se for vazia, uma cadeia de linhas ou uma cadeia multilinha. Se a conversão não puder ser executada, ocorrerá uma condição de exceção (SQLSTATE 38SUD).

Tipo de Retorno:

db2gse.ST_MultiLineString

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados variará, de acordo com sua exibição on-line

Este exemplo ilustra a utilização da função ST_ToMultiLine.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES (1, ST_Geometry ('multilinestring ((0 10 1, 0 0 3, 10 0 5),
                                           (23 43, 27 34, 35 12))', 0) ),
         (2, ST_Geometry ('linestring z (0 10 1, 0 0 3, 10 0 5)', 0) ),
         (3, ST_Geometry ('point empty', 1) ),
         (4, ST_Geometry ('multipolygon empty', 1) )
```

Na instrução SELECT a seguir, a função ST_ToMultiLine é utilizada para retornar cadeias multilinha convertidas em ST_MultiLineString a partir do tipo estático de ST_Geometry.

```
SELECT CAST( ST_AsText( ST_ToMultiLine(geometry) )
  AS VARCHAR(130) ) LINES
  FROM sample_geometries
```

Resultados:

```
LINES
----------------------------------------------------
MULTILINESTRING Z ( 0.00000000 10.00000000 1.00000000,
                    0.00000000 0.00000000 3.00000000,
                   10.00000000 0.00000000 5.00000000)
MULTILINESTRING EMPTY
MULTILINESTRING EMPTY
```

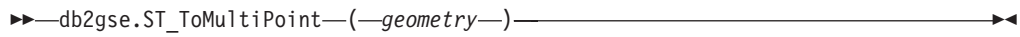
ST_ToMultiPoint

ST_ToMultiPoint utiliza uma geometria como um parâmetro de entrada e a converte em um multiponto. O multiponto resultante é representado no sistema de referência espacial da geometria especificada.

A geometria especificada deve ser vazia, um ponto ou um multiponto. Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que é convertida em um multiponto.

Uma geometria pode ser convertida em um multiponto se for vazia, um ponto ou um multiponto. Se a conversão não puder ser executada, ocorrerá uma condição de exceção (SQLSTATE 38SUD).

Tipo de Retorno:

db2gse.ST_MultiPoint

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados variará, de acordo com sua exibição on-line

Este exemplo ilustra a utilização da função ST_ToMultiPoint.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES (1, ST_Geometry ('multipoint (0 0, 0 4)', 1) ),
         (2, ST_Geometry ('point (30 40)', 1) ),
         (3, ST_Geometry ('multipolygon empty', 1) )
```

Na instrução SELECT a seguir, a função ST_ToMultiPoint é utilizada para retornar multipontos convertidos em ST_MultiPoint a partir do tipo estático de ST_Geometry.

```
SELECT CAST( ST_AsText( ST_ToMultiPoint(geometry))
  AS VARCHAR(62) ) MULTIPOINTS
FROM   sample_geometries
```

Resultados:

```
MULTIPOINTS
------------------------------------------------------------
MULTIPOINT ( 0.00000000 0.00000000, 0.00000000 4.00000000)
MULTIPOINT ( 30.00000000 40.00000000)
MULTIPOINT EMPTY
```

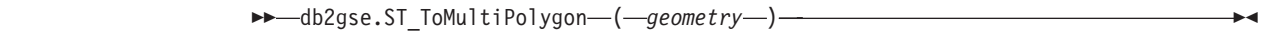
ST_ToMultiPolygon

ST_ToMultiPolygon utiliza uma geometria como um parâmetro de entrada e a converte em um multipolígono. O multipolígono resultante é representado no sistema de referência espacial da geometria especificada.

A geometria especificada deve ser vazia, um polígono ou um multipolígono. Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetro:

geometry
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que é convertida em um multipolígono.

  Uma geometria pode ser convertida em um multipolígono se for vazia, polígono ou multipolígono. Se a conversão não puder ser executada, ocorrerá uma condição de exceção (SQLSTATE 38SUD).

Tipo de Retorno:

db2gse.ST_MultiPolygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados variará, de acordo com sua exibição on-line

Este exemplo cria várias geometrias e depois utiliza ST_ToMultiPolygon para retornar multipolígonos.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES (1, ST_Geometry ('polygon ((0 0, 0 4, 5 4, 5 0, 0 0))', 1)),
         (2, ST_Geometry ('point empty', 1)),
         (3, ST_Geometry ('multipoint empty', 1))
```

Na instrução SELECT a seguir, a função ST_ToMultiPolygon é utilizada para retornar multipolígonos convertidos em ST_MultiPolygon a partir do tipo estático de ST_Geometry.

```
SELECT CAST( ST_AsText( ST_ToMultiPolygon(geometry) )
  AS VARCHAR(130) ) POLYGONS
FROM   sample_geometries
```

Resultados:

```
POLYGONS
------------------------------------------------------------------------
MULTIPOLYGON (( 0.00000000 0.00000000, 5.00000000 0.00000000,
                5.00000000 4.00000000, 0.00000000 4.00000000,
                 0.00000000 0.00000000))

MULTIPOLYGON EMPTY

MULTIPOLYGON EMPTY
```

ST_ToPoint

ST_ToPoint utiliza uma geometria como um parâmetro de entrada e a converte em um ponto. O ponto resultante é representado no sistema de referência espacial da geometria especificada.

A geometria especificada deve ser vazia ou um ponto. Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
: Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que é convertida em um ponto.

  Uma geometria pode ser convertida em um ponto se for vazia ou um ponto. Se a conversão não puder ser executada, ocorrerá uma condição de exceção (SQLSTATE 38SUD).

Tipo de Retorno:

db2gse.ST_Point

Exemplo:

Este exemplo cria três geometrias em SAMPLE_GEOMETRIES e converte cada uma em um ponto.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES (1, ST_Geometry ('point (30 40)', 1) ),
         (2, ST_Geometry ('linestring empty', 1) ),
         (3, ST_Geometry ('multipolygon empty', 1) )
```

Na instrução SELECT a seguir, a função ST_ToPoint é utilizada para retornar pontos convertidos em ST_Point a partir do tipo estático de ST_Geometry.

```
SELECT CAST( ST_AsText( ST_ToPoint(geometry) ) AS VARCHAR(35) ) POINTS
FROM   sample_geometries
```

Resultados:

```
POINTS
--------------------------------
POINT ( 30.00000000 40.00000000)
POINT EMPTY
POINT EMPTY
```

ST_ToPolygon

ST_ToPolygon utiliza uma geometria como um parâmetro de entrada e a converte em um polígono. O polígono resultante é representado no sistema de referência espacial da geometria especificada.

A geometria especificada deve ser vazia ou um polígono. Se a geometria especificada for nula, então nulo é retornado.

Esta função também pode ser chamada como um método.

Sintaxe:

Parâmetro:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que é convertida em um polígono.

Uma geometria pode ser convertida em um polígono se for vazia ou um polígono. Se a conversão não puder ser executada, ocorrerá uma condição de exceção (SQLSTATE 38SUD).

Tipo de Retorno:

db2gse.ST_Polygon

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados variará, de acordo com sua exibição on-line

Este exemplo cria três geometrias em SAMPLE_GEOMETRIES e converte cada uma em um polígono.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)

INSERT INTO sample_geometries
  VALUES (1, ST_Geometry ('polygon ((0 0, 0 4, 5 4, 5 0, 0 0))', 1) ),
         (2, ST_Geometry ('point empty', 1) ),
         (3, ST_Geometry ('multipolygon empty', 1) )
```

Na instrução SELECT a seguir, a função ST_ToPolygon é utilizada para retornar polígonos convertidos em ST_Polygon a partir do tipo estático de ST_Geometry.

```
SELECT CAST( ST_AsText( ST_ToPolygon(geometry) ) AS VARCHAR(130) ) POLYGONS
FROM   sample_geometries
```

Resultados:

```
POLYGONS
--------------------------------------------------------------------------------
POLYGON (( 0.00000000 0.00000000, 5.00000000 0.00000000,
           5.00000000 4.00000000,0.00000000 4.00000000,
            0.00000000 0.00000000))

POLYGON EMPTY

POLYGON EMPTY
```

ST_Touches

ST_Touches utiliza duas geometrias como parâmetros de entrada e retorna 1 se as geometrias especificadas se cruzarem espacialmente. Caso contrário, será retornado 0 (zero).

Duas geometrias se cruzam se os interiores das duas não fizerem interseção, mas o limite de uma das geometrias fizer interseção com o limite ou o interior da outra geometria.

Se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial.

Se as duas geometrias especificadas forem pontos ou multipontos, ou se qualquer uma das geometrias especificadas for nula ou vazia, será retornado nulo.

Sintaxe:

Parâmetros:

geometry1
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada para cruzar com *geometry2*.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que será testada para cruzar com *geometry1*.

Tipo de Retorno:

INTEGER

Exemplo:

Várias geometrias são incluídas na tabela SAMPLE_GEOMS. A função ST_Touches é utilizada para determinar quais geometrias se cruzam.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry ST_Geometry)
```

```
INSERT INTO sample_geoms
  VALUES (1, ST_Geometry('polygon ( (20 30, 30 30, 30 40, 20 40, 20 30) )' , 0) )

INSERT INTO sample_geoms
   VALUES (2, ST_Geometry('polygon ( (30 30, 30 50, 50 50, 50 30, 30 30) )' ,0) )

INSERT INTO sample_geoms
  VALUES (3, ST_Geometry('polygon ( (40 40, 40 60, 60 60, 60 40, 40 40) )' , 0) )

INSERT INTO sample_geoms
  VALUES (4, ST_Geometry('linestring( 60 60, 70 70 )' , 0) )

INSERT INTO sample_geoms
  VALUES (5, ST_Geometry('linestring( 30 30, 60 60 )' , 0) )

SELECT a.id, b.id, ST_Touches (a.geometry, b.geometry) TOUCHES
  FROM sample_geoms a, sample_geoms b
  WHERE b.id >= a.id
```

Resultados:

```
     ID          ID          TOUCHES
     ----------- ----------- -----------
               1           1           0
               1           2           1
               1           3           0
               1           4           0
               1           5           1
               2           2           0
               2           3           0
               2           4           0
               2           5           1
               3           3           0
               3           4           1
               3           5           1
               4           4           0
               4           5           1
               5           5           0
```

Referência Relacionada:

- "Funções que Utilizam Índices para Otimizar Consultas" na página 124

ST_Transform

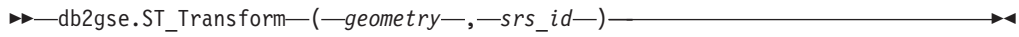
ST_Transform utiliza uma geometria e um identificador do sistema de referência espacial como parâmetros de entrada e transforma a geometria a ser representada no sistema de referência espacial especificado. As projeções e conversões entre diferentes sistemas de coordenadas são executadas e as coordenadas das geometrias são ajustadas de acordo.

A geometria somente pode ser convertida no sistema de referência espacial especificado se o sistema de referência espacial atual da geometria estiver baseado no mesmo sistema de coordenadas geográficas que o sistema de referência espacial especificado. Se nem o sistema de referência espacial atual nem o sistema de referência espacial especificado da geometria estiver baseado em um sistema de coordenadas projetadas, será executada uma projeção reversa para determinar o sistema de coordenadas geográficas que suporta o projetado.

Se a geometria especificada for nula, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

geometry
Um valor do tipo ST_Geometry ou um de seus subtipos que representa a geometria que é transformada no sistema de referência espacial identificado por *srs_id*.

srs_id Um valor do tipo INTEGER que identifica o sistema de referência espacial da geometria resultante.

Se a transformação no sistema de referência espacial especificado não puder ser executada porque o sistema de referência espacial atual da *geometry* não é compatível com o sistema de referência espacial identificado por *srs_id*, ocorrerá uma condição de exceção (SQLSTATE 38SUC).

Se *srs_id* não identificar um sistema de referência espacial listado na exibição do catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS, ocorrerá uma condição de exceção (SQLSTATE 38SU1).

Tipo de Retorno:

db2gse.ST_Geometry

Exemplos:

Os exemplos a seguir ilustram a utilização de ST_Transform para converter uma geometria de um sistema de referência espacial em outro.

Primeiro, é criado o sistema de referência espacial plano de estado com um ID 3 utilizando uma chamada para db2se.

```
db2se create_srs SAMP_DB
   -srsId 3 -srsName z3101a -xOffset 0 -yOffset 0 -xScale 1 -yScale 1
   - coordsysName NAD_1983_StatePlane_New_York_East_FIPS_3101_Feet
```

Em seguida, são incluídos pontos em:

- Tabela SAMPLE_POINTS_SP nas coordenadas planas de estado utilizando esse sistema de referência espacial.
- Tabela SAMPLE_POINTS_LL utilizando as coordenadas especificadas na latitude e longitude.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points_sp (id INTEGER, geometry ST_Point)
CREATE TABLE sample_points_ll (id INTEGER, geometry ST_Point)

INSERT INTO sample_points_sp
  VALUES (12457, ST_Point('point ( 567176.0 1166411.0)', 3) )

INSERT INTO sample_points_sp
  VALUES (12477, ST_Point('point ( 637948.0 1177640.0)', 3) )

INSERT INTO sample_points_ll
  VALUES (12457, ST_Point('point ( -74.22371600 42.03498700)', 1) )

INSERT INTO sample_points_ll
  VALUES (12477, ST_Point('point ( -73.96293200 42.06487900)', 1) )
```

Em seguida, a função ST_Transform é utilizada para converter as geometrias.

Exemplo 1:

Este exemplo converte pontos que estão nas coordenadas de latitude e longitude em coordenadas planas de estado.

```
SELECT id, CAST( ST_AsText( ST_Transform( geometry, 3) )
    AS VARCHAR(100) ) STATE_PLANE
  FROM sample_points_ll
```

Resultados:

```
ID          STATE_PLANE
----------- ----------------------------------------
      12457 POINT ( 567176.00000000 1166411.00000000)
      12477 POINT ( 637948.00000000 1177640.00000000)
```

Exemplo 2:

Este exemplo converte pontos que estão nas coordenadas planas de estado em coordenadas de latitude e longitude.

```
SELECT id, CAST( ST_AsText( ST_Transform( geometry, 1) )
    AS VARCHAR(100) ) LAT_LONG
  FROM sample_points_sp
```

Resultados:

```
ID          LAT_LONG
----------- ----------------------------------------------------------------
      12457 POINT ( -74.22371500 42.03498800)
      12477 POINT ( -73.96293100 42.06488000)
```

Referência Relacionada:

- "A Exibição do Catálogo DB2GSE.ST_SPATIAL_REFERENCE_SYSTEMS" na página 295

ST_Union

ST_Union utiliza duas geometrias como parâmetros de entrada e retorna a geometria que é a união das geometrias especificadas. A geometria resultante é representada no sistema de referência espacial da primeira geometria.

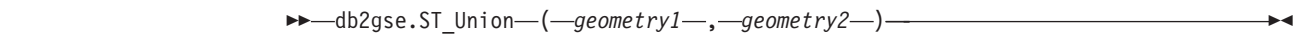
As duas geometrias devem ter a mesma dimensão.Se qualquer uma das duas geometrias especificadas for nula, será retornado nulo.

Para dados não geodésicos, se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial. Para dados geodésicos, as duas geometrias devem estar no mesmo sistema de referência espacial (SRS) geodésico.

A geometria resultante é representada no tipo espacial mais apropriado. Se puder ser representada como um ponto, cadeia de linhas ou polígono, será utilizado um desses tipos. Caso contrário, será utilizado o tipo multiponto, cadeia multilinha ou multipolígono.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

geometry1
Um valor do tipo ST_Geometry ou um de seus subtipos que é combinado com *geometry2*.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que é combinado com *geometry1*.

Restrições para dados geodésicos: As duas geometrias devem ser geodésicas e devem estar no mesmo SRS geodésico.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplos:

As seguintes instruções SQL criam e ocupam a tabela SAMPLE_GEOMS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geoms (id INTEGER, geometry, ST_Geometry)

INSERT INTO sample_geoms
  VALUES (1, ST_Geometry( 'polygon
         ((10 10, 10 20, 20 20, 20 10, 10 10) )', 0))

INSERT INTO sample_geoms
  VALUES (2, ST_Geometry( 'polygon
         ((30 30, 30 50, 50 50, 50 30, 30 30) )', 0))

INSERT INTO sample_geoms
  VALUES (3, ST_Geometry( 'polygon
         ((40 40, 40 60, 60 60, 60 40, 40 40) )', 0))

INSERT INTO sample_geoms
  VALUES (4, ST_Geometry('linestring (70 70, 80 80)', 0))

INSERT INTO sample_geoms
  VALUES (5, ST_Geometry('linestring (80 80, 100 70)', 0))
```

Nos exemplos a seguir, os resultados foram reformatados para possibilitar a leitura. Seus resultados variarão, de acordo com sua exibição.

Exemplo 1:

Este exemplo encontra a união de dois polígonos separados.

```
SELECT a.id, b.id, CAST ( ST_AsText( ST_Union( a.geometry, b.geometry) )
    AS VARCHAR (350) ) UNION
  FROM sample_geoms a, sample_geoms b
  WHERE a.id = 1 AND b.id = 2
```

Resultados:

```
ID    ID    UNION
----- ----- ----------------------------------------------------
    1     2 MULTIPOLYGON ((( 10.00000000 10.00000000, 20.00000000
                  10.00000000, 20.00000000 20.00000000, 10.00000000
                  20.00000000, 10.00000000 10.00000000))
```

```
                    (( 30.00000000 30.00000000, 50.00000000
30.00000000,50.00000000 50.00000000, 30.00000000
50.00000000,30.00000000 30.00000000)))
```

Exemplo 2:

Este exemplo encontra a união de dois polígonos que fazem interseção.

```
SELECT a.id, b.id, CAST ( ST_AsText( ST_Union( a.geometry, b.geometry) )
    AS VARCHAR (250)) UNION
  FROM sample_geoms a, sample_geoms b
  WHERE a.id = 2 AND b.id = 3
```

Resultados:

```
ID    ID    UNION
----- ----- ----------------------------------------------------
    2     3 POLYGON (( 30.00000000 30.00000000, 50.00000000
                30.00000000,50.00000000 40.00000000, 60.00000000
                40.00000000,60.00000000 60.00000000, 40.00000000
                60.00000000 40.00000000 50.00000000, 30.00000000
                50.00000000, 30.00000000 30.00000000))
```

Exemplo 3:

Localizar a união de duas cadeias de linhas.

```
SELECT a.id, b.id, CAST ( ST_AsText( ST_Union( a.geometry, b.geometry) )
    AS VARCHAR (250) ) UNION
  FROM sample_geoms a, sample_geoms b
  WHERE a.id = 4 AND b.id = 5
```

Resultados:

```
ID    ID    UNION
----- ----- ------------------------------------------------------------
    4     5 MULTILINESTRING (( 70.00000000 70.00000000, 80.00000000 80.00000000),
                ( 80.00000000 80.00000000, 100.00000000 70.00000000))
```

ST_Within

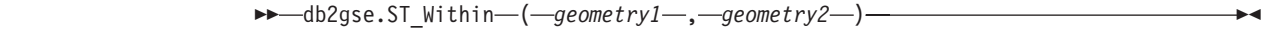
ST_Within utiliza duas geometrias como parâmetros de entrada e retorna 1 se a primeira geometria estiver totalmente dentro da segunda. Caso contrário, será retornado 0 (zero).

Se qualquer uma das geometrias especificadas for nula ou vazia, será retornado nulo.

Para dados não geodésicos, se a segunda geometria não for representada no mesmo sistema de referência espacial que a primeira geometria, ela será convertida para o outro sistema de referência espacial. Para dados geodésicos, as duas geometrias devem estar no mesmo sistema de referência espacial (SRS) geodésico.

ST_Within executa a mesma operação lógica que ST_Contains executa com os parâmetros reversos.

Sintaxe:



Parâmetros:

geometry1
Um valor do tipo ST_Geometry ou um de seus subtipos que deve ser testado para que esteja totalmente dentro de *geometry2*.

geometry2
Um valor do tipo ST_Geometry ou um de seus subtipos que deve ser testado para que esteja totalmente dentro de *geometry1*.

Restrições para dados geodésicos: As duas geometrias devem ser geodésicas e devem estar no mesmo SRS geodésico.

Tipo de Retorno:

INTEGER

Exemplos:

Estes exemplos ilustram a utilização da função ST_Within. As geometrias são criadas e inseridas em três tabelas, SAMPLE_POINTS, SAMPLE_LINES e SAMPLE_POLYGONS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)
CREATE TABLE sample_lines (id INTEGER, line ST_LineString)
CREATE TABLE sample_polygons (id INTEGER, geometry ST_Polygon)

INSERT INTO sample_points (id, geometry)
  VALUES (1, ST_Point (10, 20, 1) ),
         (2, ST_Point ('point (41 41)', 1) )

INSERT INTO sample_lines (id, line)
  VALUES (10, ST_LineString ('linestring (1 10, 3 12, 10 10)', 1) ),
         (20, ST_LineString ('linestring (50 10, 50 12, 45 10)', 1) )

INSERT INTO sample_polygons (id, geometry)
  VALUES (100, ST_Polygon ('polygon (( 0 0, 0 40, 40 40, 40 0, 0 0))', 1) )
```

Exemplo 1:

Este exemplo encontra pontos na tabela SAMPLE_POINTS que estão nos polígonos na tabela SAMPLE_POLYGONS.

```
SELECT a.id POINT_ID_WITHIN_POLYGONS
  FROM sample_points a, sample_polygons b
  WHERE ST_Within( b.geometry, a.geometry) = 0
```

Resultados:

```
POINT_ID_WITHIN_POLYGONS
----------------------
                     2
```

Exemplo 2:

Este exemplo encontra cadeias de linhas da tabela SAMPLE_LINES que estão nos polígonos da tabela SAMPLE_POLYGONS.

```
SELECT a.id LINE_ID_WITHIN_POLYGONS
  FROM sample_lines a, sample_polygons b
  WHERE ST_Within( b.geometry, a.geometry) = 0
```

Resultados:

```
LINE_ID_WITHIN_POLYGONS
----------------------
                     1
```

Referência Relacionada:

- "ST_Contains" na página 363

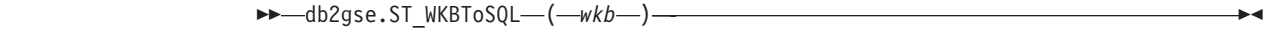
ST_WKBToSQL

ST_WKBToSQL utiliza uma representação binária reconhecida de uma geometria e retorna a geometria correspondente. O sistema de referência espacial com o identificador 0 (zero) é utilizado para a geometria resultante.

Se representação binária reconhecida especificada for nula, então nulo é retornado.

ST_WKBToSQL(*wkb*) fornece o mesmo resultado que ST_Geometry(*wkb*,0). A utilização da função ST_Geometry é recomendada sobre a utilização de ST_WKBToSQL por sua flexibilidade: ST_Geometry utiliza formatos adicionais de entrada, além da representação binária reconhecida.

Sintaxe:



Parâmetro:

wkb Um valor do tipo BLOB(2G) que contém a representação binária reconhecida da geometria resultante.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra a utilização da função ST_WKBToSQL. Primeiro, as geometrias são armazenadas na tabela SAMPLE_GEOMETRIES em sua coluna GEOMETRY. Em seguida, suas representações binárias reconhecidas são armazenadas na coluna WKB utilizando a função ST_AsBinary na instrução UPDATE. Por último, a função ST_WKBToSQL é utilizada para retornar as coordenadas das geometrias na coluna WKB.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries
  (id INTEGER, geometry ST_Geometry, wkb BLOB(32K) )

INSERT INTO sample_geometries (id, geometry)
  VALUES (10, ST_Point ( 'point (44 14)', 0 ) ),
         (11, ST_Point ( 'point (24 13)', 0 ) ),
         (12, ST_Polygon ('polygon ((50 20, 50 40, 70 30, 50 20))', 0 ) )
UPDATE sample_geometries AS temp_correlated
  SET wkb = ST_AsBinary(geometry)
  WHERE id = temp_correlated.id
```

Utilize esta instrução SELECT para ver as geometrias na coluna WKB.

```
SELECT id, CAST( ST_AsText( ST_WKBToSQL(wkb) ) AS VARCHAR(120) ) GEOMETRIES
  FROM sample_geometries
```

Resultados:

```
ID          GEOMETRIES
----------- ------------------------------------------------------------
         10 POINT ( 44.00000000 14.00000000)
         11 POINT ( 24.00000000 13.00000000)
         12 POLYGON (( 50.00000000 20.00000000, 70.00000000 30.00000000,
                       50.00000000 40.00000000, 50.00000000 20.00000000))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_Geometry" na página 397
- "ST_WKTToSQL" na página 505

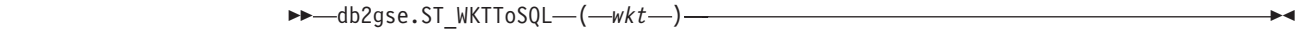
ST_WKTToSQL

ST_WKTToSQL utiliza uma representação de texto reconhecida de uma geometria e retorna a geometria correspondida. O sistema de referência espacial com o identificador 0 (zero) é utilizado para a geometria resultante.

Se a representação de texto reconhecida for nula, então nulo é retornado.

ST_WKTToSQL(*wkt*) fornece o mesmo resultado que ST_Geometry(*wkt*,0). A utilização da função ST_Geometry é recomendada sobre a utilização de ST_WKTToSQL por sua flexibilidade: ST_Geometry utiliza formatos adicionais de entrada, além da representação de texto reconhecida.

Sintaxe:



Parâmetro:

wkt Um valor do tipo CLOB(2G) que contém a representação de texto reconhecida da geometria resultante.

Tipo de Retorno:

db2gse.ST_Geometry

Exemplo:

No exemplo a seguir, as linhas de resultados foram formatadas novamente para possibilidade de leitura. O espaçamento em seus resultados irão variar de acordo com sua exibição on-line.

Este exemplo ilustra como ST_WKTToSQL pode criar e inserir geometrias utilizando suas representações de texto reconhecidas.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_geometries (id INTEGER, geometry ST_Geometry)
```

```
INSERT INTO sample_geometries
  VALUES (10, ST_WKTToSQL( 'point (44 14)' ) ),
         (11, ST_WKTTSQL ( 'point (24 13)' ) ),
         (12, ST_WKTToSQL ('polygon ((50 20, 50 40, 70 30, 50 20))' ) )
```

Esta instrução SELECT retorna as geometrias que foram inseridas.

```
SELECT id, CAST( ST_AsText(geometry) AS VARCHAR(120) ) GEOMETRIES
FROM   sample_geometries
```

Resultados:

```
ID          GEOMETRIES
----------- ------------------------------------------------------------
         10 POINT ( 44.00000000 14.00000000)
         11 POINT ( 24.00000000 13.00000000)
         12 POLYGON (( 50.00000000 20.00000000, 70.00000000 30.00000000,
                       50.00000000 40.00000000, 50.00000000 20.00000000))
```

Conceitos Relacionados:

- "Dados Espaciais e Geodésicos" na página 4

Referência Relacionada:

- "ST_Geometry" na página 397
- "ST_WKBToSQL" na página 504

ST_X

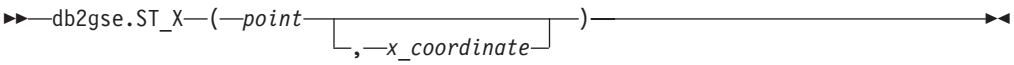
ST_X utiliza:

- Um ponto como parâmetro de entrada e retorna sua coordenada X
- Um ponto e uma coordenada X e retorna o próprio ponto com sua coordenada X definida como o valor especificado

Se o ponto especificado for nulo ou vazio, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

point Um valor do tipo ST_Point para o qual a coordenada X é retornada ou modificada.

x_coordinate
Um valor do tipo DOUBLE que representa a nova coordenada X para *point*.

Tipos de Retorno:

- DOUBLE, se *x_coordinate* não for especificado
- db2gse.ST_Point, se *x_coordinate* for especificado

Exemplos:

Estes exemplos ilustram a utilização da função ST_X. As geometrias são criadas e inseridas na tabela SAMPLE_POINTS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)

INSERT INTO sample_points (id, geometry)
  VALUES (1, ST_Point (2, 3, 32, 5, 1) ),
         (2, ST_Point (4, 5, 20, 4, 1) ),
         (3, ST_Point (3, 8, 23, 7, 1) )
```

Exemplo 1:

Este exemplo encontra as coordenadas X dos pontos na tabela.

```
SELECT id, ST_X (geometry) X_COORD
  FROM sample_points
```

Resultados:

```
ID         X_COORD
---------- ----------------------
         1 +2.00000000000000E+000
         2 +4.00000000000000E+000
         3 +3.00000000000000E+000
```

Exemplo 2:

Este exemplo retorna um ponto com sua coordenada X definida como 40.

```
SELECT id, CAST( ST_AsText( ST_X (geometry, 40)) AS VARCHAR(60) )
  X_40
  FROM sample_points
  WHERE id=3
```

Resultados:

```
ID         X_40
------ ------------------------------------------------------------
     3 POINT ZM ( 40.00000000 8.00000000 23.00000000 7.00000000)
```

Referência Relacionada:

- "ST_M" na página 425
- "ST_Y" na página 507
- "ST_Z" na página 509

ST_Y

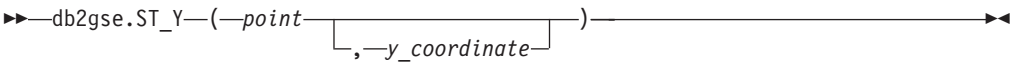
ST_Y utiliza:

- Um ponto como parâmetro de entrada e retorna sua coordenada Y
- Um ponto e uma coordenada Y e retorna o próprio ponto com sua coordenada Y definida como o valor especificado

Se o ponto especificado for nulo ou vazio, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

point Um valor do tipo ST_Point para o qual a coordenada Y é retornada ou modificada.

y_coordinate
Um valor do tipo DOUBLE que representa a nova coordenada Y para *point*.

Tipos de Retorno:

- DOUBLE, se *y_coordinate* não for especificado
- db2gse.ST_Point, se *y_coordinate* for especificado

Exemplos:

Estes exemplos ilustram a utilização da função ST_Y. As geometrias são criadas e inseridas na tabela SAMPLE_POINTS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)

INSERT INTO sample_points (id, geometry)
  VALUES (1, ST_Point (2, 3, 32, 5, 1) ),
         (2, ST_Point (4, 5, 20, 4, 1) ),
         (3, ST_Point (3, 8, 23, 7, 1) )
```

Exemplo 1:

Este exemplo encontra as coordenadas Y dos pontos na tabela.

```
SELECT id, ST_Y (geometry) Y_COORD
  FROM sample_points
```

Resultados:

```
ID         Y_COORD
---------- ----------------------
         1 +3.00000000000000E+000
         2 +5.00000000000000E+000
         3 +8.00000000000000E+000
```

Exemplo 2:

Este exemplo retorna um ponto com sua coordenada Y definida como 40.

```
SELECT id, CAST( ST_AsText( ST_Y (geometry, 40)) AS VARCHAR(60) )
  Y_40
  FROM sample_points
  WHERE id=3
```

Resultados:

```
ID         Y_40
-------- ------------------------------------------------------
     3 POINT ZM ( 3.00000000 40.00000000 23.00000000 7.00000000)
```

Referência Relacionada:

- "ST_M" na página 425
- "ST_X" na página 506
- "ST_Z" na página 509

ST_Z

ST_Z utiliza:

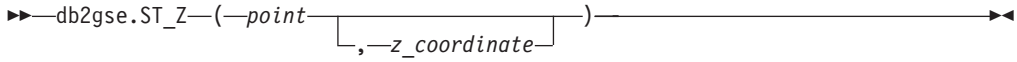
- Um ponto como parâmetro de entrada e retorna sua coordenada Z
- Um ponto e uma coordenada Z e retorna o próprio ponto com sua coordenada Z definida como o valor especificado, mesmo que o ponto especificado não tenha nenhuma coordenada Z existente.

Se a coordenada Z especificada for nula, a coordenada Z será removida do ponto.

Se o ponto especificado for nulo ou vazio, será retornado nulo.

Esta função também pode ser chamada como um método.

Sintaxe:



Parâmetros:

point Um valor do tipo ST_Point para o qual a coordenada Z é retornada ou modificada.

z_coordinate
Um valor do tipo DOUBLE que representa a nova coordenada Z para *point*.

Se *z_coordinate* for nulo, a coordenada Z será removida de *point*.

Tipos de Retorno:

- DOUBLE, se *z_coordinate* não for especificado
- db2gse.ST_Point, se *z_coordinate* for especificado

Exemplos:

Estes exemplos ilustram a utilização da função ST_Z. As geometrias são criadas e inseridas na tabela SAMPLE_POINTS.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)

INSERT INTO sample_points (id, geometry)
  VALUES (1, ST_Point (2, 3, 32, 5, 1) ),
         (2, ST_Point (4, 5, 20, 4, 1) ),
         (3, ST_Point (3, 8, 23, 7, 1) )
```

Exemplo 1:

Este exemplo encontra as coordenadas Z dos pontos na tabela.

```
SELECT id, ST_Z (geometry) Z_COORD
  FROM sample_points
```

Resultados:

```
ID         Z_COORD
---------- ----------------------
         1 +3.20000000000000E+001
         2 +2.00000000000000E+001
         3 +2.30000000000000E+001
```

Exemplo 2:

Este exemplo retorna um ponto com sua coordenada Z definida como 40.

```
SELECT id, CAST( ST_AsText( ST_Z (geometry, 40)) AS VARCHAR(60) )
  Z_40
  FROM sample_points
  WHERE id=3
```

Resultados:

```
ID         Z_40
------ ------------------------------------------------------
     3 POINT ZM ( 3.00000000 8.00000000 40.00000000 7.00000000)
```

Referência Relacionada:

- "ST_M" na página 425
- "ST_X" na página 506
- "ST_Y" na página 507

Agregado de União

Um agregado de união é a combinação das funções ST_BuildUnionAggr e ST_GetAggrResult. Essa combinação agrega uma coluna de geometrias em uma tabela em uma única geometria, construindo a união.

Se todas as geometrias a serem combinadas na união forem nulas, será retornado nulo. Se cada uma das geometrias a serem combinadas na união forem nulas ou vazias, será retornada uma geometria vazia do tipo ST_Point.

A função ST_BuildUnionAggr também pode ser chamada como um método.

Sintaxe:

Parâmetros:

geometries
Uma coluna em uma tabela do tipo ST_Geometry ou um dos seus subtipos e representa todas as geometrias que devem ser combinadas em uma união.

Tipo de Retorno:

db2gse.ST_Geometry

Restrições:

Não é possível construir o agregado de união de uma coluna espacial em uma tabela em qualquer uma das seguintes situações:

- Em ambientes com MPP (processamento paralelo massivo)
- Se uma cláusula GROUP BY for utilizada na seleção
- Se você utilizar uma função diferente da função agregada do DB2 MAX

Exemplo:

No exemplo a seguir, as linhas de resultados foram reformatadas para facilitar a leitura. O espaçamento nos resultados varia de acordo com a exibição on-line.

Este exemplo ilustra como um agregado de união pode ser utilizado para combinar um conjunto de pontos em multipontos. Diversos pontos são incluídos na tabela SAMPLE_POINTS. As funções ST_GetAggrResult e ST_BuildUnionAggr são utilizadas para construir a união dos pontos.

```
SET CURRENT FUNCTION PATH = CURRENT FUNCTION PATH, db2gse
CREATE TABLE sample_points (id INTEGER, geometry ST_Point)

INSERT INTO sample_points
  VALUES (1, ST_Point (2, 3, 1) )
INSERT INTO sample_points
  VALUES (2, ST_Point (4, 5, 1) )
INSERT INTO sample_points
  VALUES (3, ST_Point (13, 15, 1) )
INSERT INTO sample_points
  VALUES (4, ST_Point (12, 5, 1) )
INSERT INTO sample_points
  VALUES (5, ST_Point (23, 2, 1) )
INSERT INTO sample_points
  VALUES (6, ST_Point (11, 4, 1) )

SELECT CAST (ST_AsText(
          ST_GetAggrResult( MAX( ST_BuildUnionAggregate (geometry) ) ))
       AS VARCHAR(160)) POINT_AGGREGATE
  FROM sample_points
```

Resultados:

```
POINT_AGGREGATE
-------------------------------------------------------
MULTIPOINT ( 2.00000000 3.00000000, 4.00000000 5.00000000,
            11.00000000 4.00000000, 12.00000000 5.00000000,
            13.00000000 15.00000000, 23.00000000 2.00000000)
```

Agregado de União

Capítulo 24. Grupos de Transformação

Grupos de Transformação

O Spatial Extender fornece quatro grupos de transformação que são utilizados para transferir geometrias entre o servidor DB2 e um aplicativo cliente. Esses grupos de transformação acomodam os seguintes formatos de troca de dados:

- Representação WKT (Well-Known Text)
- Representação WKB (Well-Known Binary)
- Representação de Formatos ESRI
- GML (Geography Markup Language)

Quando os dados são recuperados de uma tabela que contém uma coluna espacial, os dados dessa coluna são transformados em um tipo de dados CLOB(2G) ou BLOB(2G), dependendo de você ter indicado se os dados transformados deviam ser representados em formato binário ou de texto. Também é possível utilizar os grupos de transformação para transferir dados espaciais no banco de dados.

Para selecionar qual grupo de transformação deve ser utilizado quando os dados são transferidos, utilize a instrução SET CURRENT DEFAULT TRANSFORM GROUP para modificar o registro especial CURRENT DEFAULT TRANSFORM GROUP do DB2. O DB2 utiliza o valor desse registro especial para determinar quais funções de transformação devem ser chamadas para executar as conversões necessárias.

Os grupos de transformação podem simplificar a programação dos aplicativos. Em vez de utilizar explicitamente as funções de conversão nas instruções SQL, você pode especificar um grupo de transformação, que permite que o DB2 cuide dessa tarefa.

Conceitos Relacionados:

- "Grupo de transformação ST_WellKnownText" na página 513
- "Grupo de transformação ST_WellKnownBinary" na página 514
- "Grupo de transformação ST_Shape" na página 516
- "Grupo de transformação ST_GML" na página 517

Grupo de transformação ST_WellKnownText

Você pode utilizar o grupo de transformação ST_WellKnownText para transmitir dados de e para o DB2® utilizando a representação de WKT (Well-Known Text).

Ao ligar um valor do servidor de banco de dados ao cliente, a mesma função fornecida por ST_AsText() é utilizada para converter uma geometria na representação de WKT. Quando a representação de texto reconhecido de uma geometria é transferida para o servidor de banco de dados, a função ST_Geometry(CLOB) é utilizada implicitamente para executar as conversões em um valor ST_Geometry. Utilizar o grupo de transformação para ligar valores ao DB2 faz com que as geometrias sejam representadas no sistema de referência espacial com o identificador numérico 0 (zero).

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram reformatadas para facilitar a leitura. O espaçamento nos resultados pode variar de acordo com a exibição on-line.

Exemplo 1:

O script SQL a seguir mostra como utilizar o grupo de transformação ST_WellKnownText para recuperar uma geometria em sua representação de texto reconhecido sem utilizar a função explícita ST_AsText.

```
CREATE TABLE transforms_sample (
   id   INTEGER,
   geom  db2gse.ST_Geometry)

INSERT
   INTO   transforms_sample
   VALUES (1, db2gse.ST_LineString('linestring
          (100 100, 200 100)', 0))

SET CURRENT DEFAULT TRANSFORM GROUP = ST_WellKnownText

SELECT id, geom
  FROM   transforms_sample
  WHERE  id = 1
```

Resultados:

```
ID   GEOM
---  ------------------------------------------------------------------
  1  LINESTRING ( 100.00000000 100.00000000, 200.00000000 100.00000000)
```

Exemplo 2:

O código C a seguir mostra como utilizar o grupo de transformação ST_WellKnownText para inserir geometrias utilizando a função explícita ST_Geometry para a variável de host `wkt_buffer`, que tem o tipo CLOB e contém a representação de texto reconhecido do ponto (10 10) que deve ser inserido.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS db2gse.ST_Geometry AS CLOB(1000) wkt_buffer;
EXEC SQL END DECLARE SECTION;

// definir o grupo de transformação de todas as instruções SQL subseqüentes
EXEC SQL
   SET CURRENT DEFAULT TRANSFORM GROUP = ST_WellKnownText;

id = 100;
strcpy(wkt_buffer.data, "point ( 10 10 )");
wkt_buffer.length = strlen(wkt_buffer.data);

// inserir o ponto utilizando WKT na coluna do tipo ST_Geometry
EXEC SQL
   INSERT
    INTO   transforms_sample(id, geom)
    VALUES (:id, :wkt_buffer);
```

Grupo de transformação ST_WellKnownBinary

Utilize o grupo de transformação ST_WellKnownBinary para transmitir dados de e para o DB2® utilizando a representação de WKB (binário reconhecido).

Ao ligar um valor do servidor de banco de dados ao cliente, a mesma função fornecida por ST_AsBinary() é utilizada para converter uma geometria na representação de WKB. Quando a representação de binário reconhecido de uma geometria é transferida para o servidor de banco de dados, a função ST_Geometry(BLOB) é utilizada implicitamente para executar as conversões em um valor ST_Geometry. Utilizar o grupo de transformação para ligar valores ao DB2 faz com que as geometrias sejam representadas no sistema de referência espacial com o identificador numérico 0 (zero).

Exemplo:

Nos exemplos a seguir, as linhas de resultados foram reformatadas para facilitar a leitura. O espaçamento nos resultados pode variar de acordo com a exibição on-line.

Exemplo 1:

O script SQL a seguir mostra como utilizar o grupo de transformação ST_WellKnownBinary para recuperar uma geometria em sua representação de binário reconhecido sem utilizar a função explícita ST_AsBinary.

```
CREATE TABLE transforms_sample (
   id   INTEGER,
   geom  db2gse.ST_Geometry)

INSERT
   INTO transforms_sample
   VALUES ( 1, db2gse.ST_Polygon('polygon ((10 10, 20 10, 20 20,
             10 20, 10 10))', 0))

SET CURRENT DEFAULT TRANSFORM GROUP = ST_WellKnownBinary

SELECT id, geom
   FROM transforms_sample
   WHERE id = 1
```

Resultados:

```
ID    GEOM
----   ------------------------------------------------
   1   x'0103000000010000000500000000000000000244000
       0000000002440000000000000244000000000000003440
       0000000000003440000000000000344000000000000034
       4000000000000024400000000000002440000000000000
       2440'
```

Exemplo 2:

O código C a seguir mostra como utilizar o grupo de transformação ST_WellKnownBinary para inserir geometrias utilizando a função explícita ST_Geometry para a variável de host `wkb_buffer`, que tem o tipo BLOB e contém a representação de binário reconhecido de uma geometria que deve ser inserida.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS db2gse.ST_Geometry AS BLOB(1000) wkb_buffer;
EXEC SQL END DECLARE SECTION;

// definir o grupo de transformação de todas as instruções SQL subseqüentes
EXEC SQL
   SET CURRENT DEFAULT TRANSFORM GROUP = ST_WellKnownBinary;

// inicializar variáveis do host
```

```
...
// inserir geometria utilizando WKB na coluna do tipo ST_Geometry

EXEC SQL
   INSERT
    INTO transforms_sample(id, geom)
    VALUES ( :id, :wkb_buffer );
```

Grupo de transformação ST_Shape

Utilize o grupo de transformação ST_Shape para transmitir dados de e para o DB2® utilizando a representação de formato ESRI.

Ao ligar um valor do servidor de banco de dados ao cliente, a mesma função fornecida por ST_AsShape() é utilizada para converter uma geometria em sua representação de forma. Ao transferir a representação de forma de uma geometria para o servidor de banco de dados, a função ST_Geometry(BLOB) é utilizada implicitamente para executar as conversões em um valor ST_Geometry. Utilizar o grupo de transformação para ligar valores ao DB2 faz com que as geometrias sejam representadas no sistema de referência espacial com o identificador numérico 0 (zero).

Exemplos:

Nos exemplos a seguir, as linhas de resultados foram reformatadas para facilitar a leitura. O espaçamento nos resultados pode variar de acordo com a exibição on-line.

Exemplo 1:

O script SQL a seguir mostra como o grupo de transformação ST_Shape pode ser utilizado para recuperar uma geometria em sua representação de forma sem utilizar a função explícita ST_AsShape.

```
CREATE TABLE transforms_sample (
   id INTEGER,
   geom  db2gse.ST_Geometry)

INSERT
   INTO transforms_sample
   VALUES ( 1, db2gse.ST_Point(20.0, 30.0, 0) )

SET CURRENT DEFAULT TRANSFORM GROUP = ST_Shape

SELECT id, geom
   FROM transforms_sample
   WHERE id = 1
```

Resultados:

```
ID    GEOM
----   ----------------------------------------------
   1    x'0100000000000000000034400000000000003E40'
```

Exemplo 2:

O código C a seguir mostra como utilizar o grupo de transformação ST_Shape para inserir geometrias utilizando a função explícita ST_Geometry para a variável de host `shape_buffer`, que tem o tipo BLOB e contém a representação de forma de uma geometria que deve ser inserida.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS db2gse.ST_Geometry AS BLOB(1000) shape_buffer;
EXEC SQL END DECLARE SECTION;

// definir o grupo de transformação de todas as instruções SQL subseqüentes
EXEC SQL
   SET CURRENT DEFAULT TRANSFORM GROUP = ST_Shape;

// inicializar variáveis do host
...

SET CURRENT DEFAULT TRANSFORM GROUP = ST_Shape;

// inserir geometria utilizando representação de forma na coluna do tipo ST_Geometry
EXEC SQL
     INSERT
    INTO transforms_sample(id, geom)
    VALUES ( :id, :shape_buffer );
```

Grupo de transformação ST_GML

Utilize o grupo de transformação ST_GML para transmitir dados de e para o DB2® utilizando GML (geography markup language).

Ao ligar um valor do servidor de banco de dados ao cliente, a mesma função fornecida por ST_AsGML() é utilizada para converter uma geometria em sua representação GML. Quando a representação GML de uma geometria é transferida para o servidor de banco de dados, a função ST_Geometry(CLOB) é utilizada implicitamente para executar as conversões em um valor ST_Geometry. Utilizar o grupo de transformação para ligar valores ao DB2 faz com que as geometrias sejam representadas no sistema de referência espacial com o identificador numérico 0 (zero).

Exemplos:

Nos exemplos a seguir, as linhas de resultados foram reformatadas para facilitar a leitura. O espaçamento nos resultados pode variar de acordo com a exibição on-line.

Exemplo 1:

O script SQL a seguir mostra como o grupo de transformação ST_GML pode ser utilizado para recuperar uma geometria em sua representação GML sem utilizar a função explícita ST_AsGML.

```
CREATE TABLE transforms_sample (
   id  INTEGER,
   geom  db2gse.ST_Geometry)

INSERT
  INTO  transforms_sample
  VALUES ( 1, db2gse.ST_Geometry('multipoint z (10 10
           3, 20 20 4, 15 20 30)', 0) )
  SET CURRENT DEFAULT TRANSFORM GROUP = ST_GML

SELECT id, geom
FROM  transforms_sample
WHERE  id = 1
```

Resultados:

```
ID    GEOM
-----  ------------------------------------------------------------
    1 <gml:MultiPoint srsName=UNSPECIFIED><gml:PointMember>
      <gml:Point><gml:coord><gml:X>10</gml:X>
      <gml:Y>10</gml:Y><gml:Z>3</gml:Z>
      </gml:coord></gml:Point></gml:PointMember>
      <gml:PointMember><gml:Point><gml:coord>
      <gml:X>20</gml:X><gml:Y>20</gml:Y>
      <gml:Z>4</gml:Z></gml:coord></gml:Point>
      </gml:PointMember><gml:PointMember><gml:Point>
      <gml:coord><gml:X>15</gml:X><gml:Y>20
      </gml:Y><gml:Z>30</gml:Z></gml:coord>
      </gml:Point></gml:PointMember></gml:MultiPoint>
```

Exemplo 2:

O código C a seguir mostra como utilizar o grupo de transformação ST_GML para inserir geometrias sem utilizar a função explícita ST_Geometry para a variável de host gml_buffer, que tem o tipo CLOB e contém a representação GML do ponto (20 ,20) que deve ser inserido.

```
EXEC SQL BEGIN DECLARE SECTION;
   sqlint32 id = 0;
   SQL TYPE IS db2gse.ST_Geometry AS CLOB(1000) gml_buffer;
EXEC SQL END DECLARE SECTION;

// definir o grupo de transformação de todas as instruções SQL subseqüentes
EXEC SQL
   SET CURRENT DEFAULT TRANSFORM GROUP = ST_GML;
 id = 100;
strcpy(gml_buffer.data, "<gml:point><gml:coord>"
   "<gml:X>20</gml:X> <gml:Y>20</gml:Y></gml:coord></gml:point>");

// inicializar variáveis do host
wkt_buffer.length = strlen(gml_buffer.data);

// inserir o ponto utilizando WKT na coluna do tipo ST_Geometry
EXEC SQL
   INSERT
   INTO transforms_sample(id, geom)
   VALUES ( :id, :gml_buffer );
```

Capítulo 25. Formatos de Dados Suportados

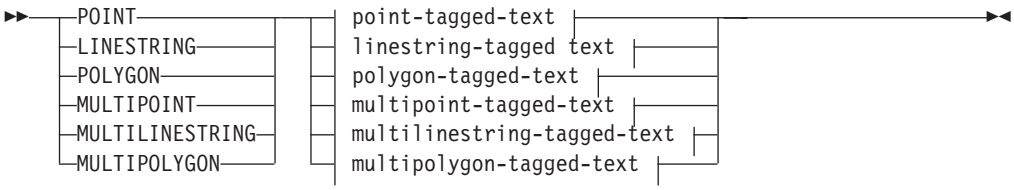
Este capítulo descreve os formatos de dados espaciais padrão da indústria que podem ser utilizados com o DB2 Spatial Extender. Consulte "Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados" na página 299 para obter informações sobre funções que aceitam e geram esses formatos. Consulte "Sobre Importação e Exportação de Dados Espaciais" na página 87 para obter informações sobre como importar e exportar arquivos contendo estes formatos. São descritos os quatro formatos de dados espaciais a seguir:

- Representação WKT (Well-Known Text)
- Representação WKB (Well-Known Binary)
- Representação de Formatos
- Representação GML (Geography Markup Language)

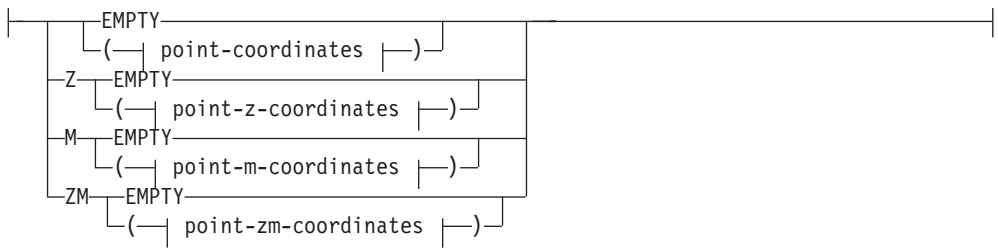
Representação WKT (Well-Known Text)

A especificação do OpenGIS Consortium ″Simple Features for SQL″ define a representação de texto reconhecida para trocar dados de geometrias em formato ASCII. Esta representação também é referida pelo padrão ISO ″SQL/MM Part: 3 Spatial″. Consulte ″Funções espaciais que convertem geometrias em e a partir de formatos de troca de dados″ para obter informações sobre funções que aceitam e geram dados WKT.

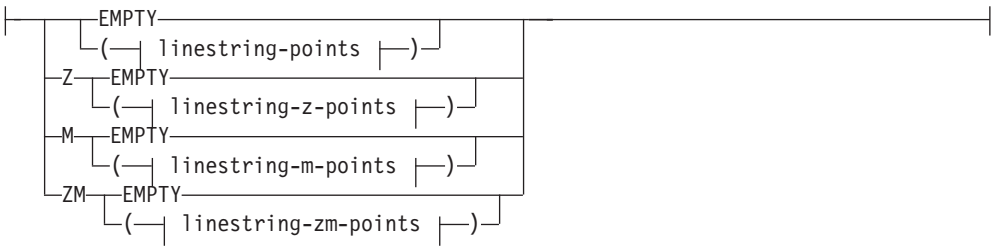
A representação de texto reconhecida de uma geometria é definida conforme a seguir:



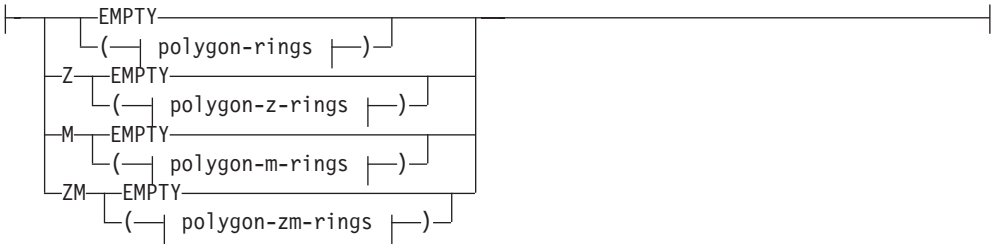
point-tagged-text:



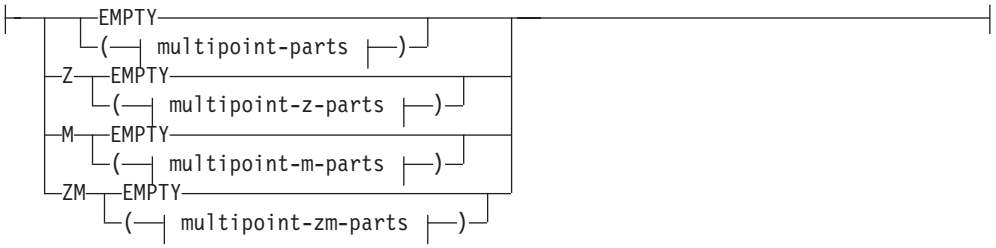
linestring-tagged-text:



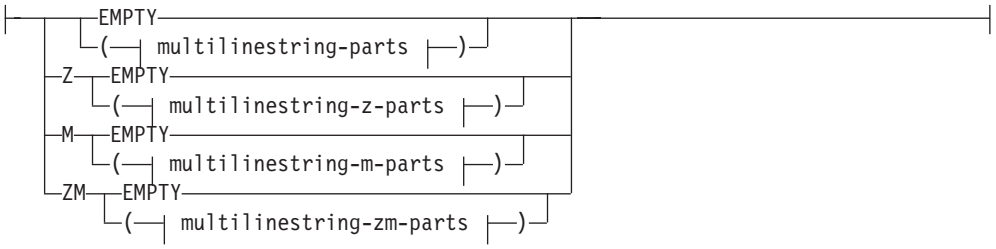
polygon-tagged-text:



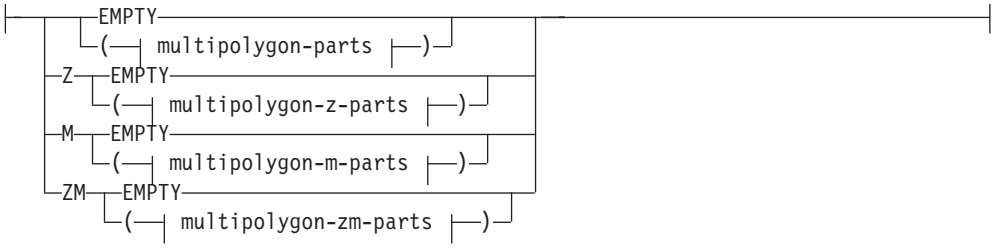
multipoint-tagged-text:



multilinestring-tagged-text:

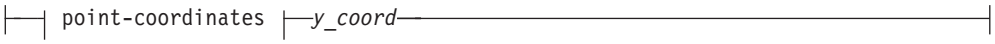


multipolygon-tagged-text:



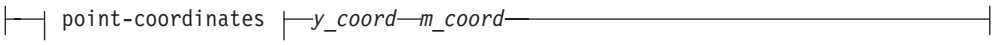
point-coordinates:

point-z-coordinates:

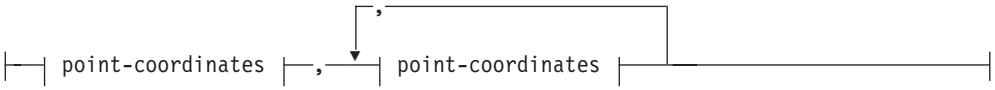


point-m-coordinates:

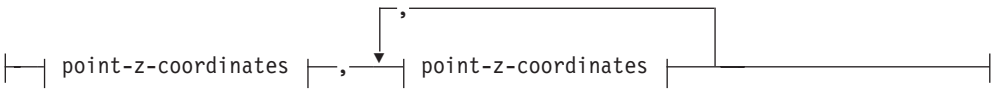
point-zm-coordinates:



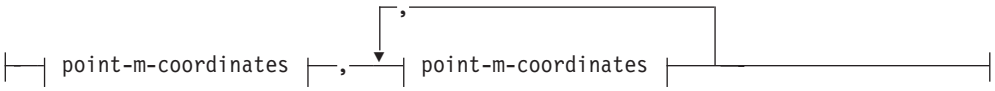
linestring-points:



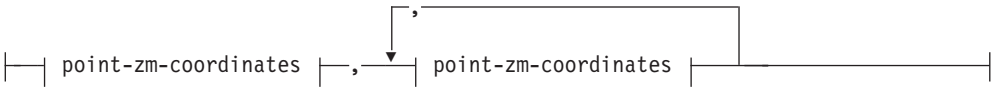
linestring-z-points:



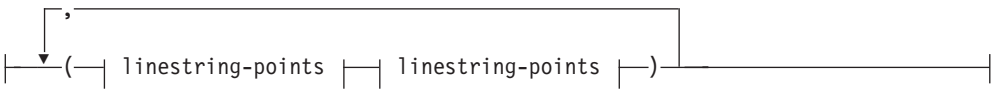
linestring-m-points:



linestring-zm-points:



polygon-rings:



polygon-z-rings:

polygon-m-rings:

polygon-zm-rings:

multipoint-parts:

multipoint-z-parts:

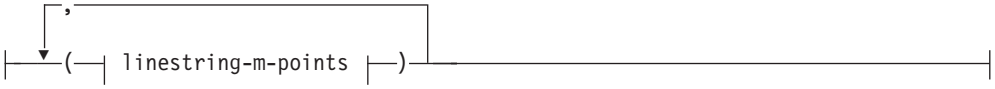
multipoint-m-parts:

multipoint-zm-parts:

multilinestring-parts:

multilinestring-z-parts:

multilinestring-m-parts:



multilinestring-zm-parts:

multipolygon-parts:

multipolygon-z-parts:

multipolygon-m-parts:

multipolygon-zm-parts:

Parâmetros:
x_coord

Um valor numérico (fixo, inteiro ou ponto flutuante) que representa a coordenada X de um ponto.

y_coord

Um valor numérico (fixo, inteiro ou ponto flutuante) que representa a coordenada Y de um ponto.

z_coord

Um valor numérico (fixo, inteiro ou ponto flutuante) que representa a coordenada Z de um ponto.

m_coord

Um valor numérico (fixo, inteiro ou ponto flutuante) que representa a coordenada M (medida) de um ponto.

Se a geometria estiver vazia, a palavra-chave EMPTY terá de ser especificada em vez da lista de coordenadas. A palavra-chave EMPTY não deve ser incorporada na lista de coordenadas

A tabela a seguir fornece alguns exemplos de possíveis representações de texto.

Tabela 57. Tipos de Figura Geométrica e Suas Representações de Texto

Tipo de Figura Geométrica	Representação WKT	Comentário
ponto	POINT EMPTY	ponto vazio
ponto	POINT (10.05 10.28)	point
ponto	POINT Z(10.05 10.28 2.51)	ponto com coordenada Z
ponto	POINT M(10.05 10.28 4.72)	ponto com coordenada M
ponto	POINT ZM(10.05 10.28 2.51 4.72)	ponto com coordenada Z e coordenada M
cadeia de linha	LINESTRING EMPTY	cadeia de linha vazia
polígono	POLYGON ((10 10, 10 20, 20 20, 20 15, 10 10))	polígono
multiponto	MULTIPOINT Z(10 10 2, 20 20 3)	multiponto com coordenadas Z
cadeia de linhas múltiplas	MULTILINESTRING M((310 30 1, 40 30 20, 50 20 10)(10 10 0, 20 20 1))	cadeia multilinha com coordenadas M
multipolígono	MULTIPOLYGON ZM(((1 1 1 1, 1 2 3 4, 2 2 5 6, 2 1 7 8, 1 1 1 1)))	multipolígono com coordenadas Z e coordenadas M

Referência Relacionada:

- "Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados" na página 299

Representação WKB (Well-Known Binary)

Esta seção descreve a representação binária reconhecida para geometrias.

A especificação do OpenGIS Consortium ″Simple Features for SQL″ define a representação binária reconhecida. Esta representação também é definida pelo padrão International Organization for Standardization (ISO) ″SQL/MM Part: 3 Spatial″. Consulte a seção de referência relacionada no final deste tópico para obter informações sobre funções que aceitam e geram WKB.

O bloco de construção básica para representações binárias reconhecidas é o fluxo de bytes para um ponto, que consiste em dois valores duplos. Os fluxos de bytes para outras figuras geométricas são construídos através de fluxos de bytes para figuras geométricas que já estejam definidas.

O exemplo a seguir ilustra o bloco de construção básica para representações binárias reconhecidas.

```
// Definições do Tipo Básico
// byte : 1 byte
// uint32 : inteiro não-assinado de 32 bits (4 bytes)
// double : número de precisão dupla (8 bytes)

// Construindo Blocos : Ponto, Anel Linear

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=wkbPoint
  Point    point;
};
WKBLineString {
  byte     byteOrder;
  uint32   wkbType;     // 2=wkbLineString
  uint32   numPoints;
  Point    points[numPoints];
};

WKBPolygon    {
  byte               byteOrder;
  uint32            wkbType;     // 3=wkbPolygon
  uint32            numRings;
  LinearRing        rings[numRings];
};
WKBMultiPoint    {
  byte               byteOrder;
  uint32            wkbType;     // 4=wkbMultipoint
  uint32            num_wkbPoints;
  WKBPoint            WKBPoints[num_wkbPoints];
};
WKBMultiLineString    {
  byte              byteOrder;
  uint32            wkbType;     // 5=wkbMultiLineString
  uint32            num_wkbLineStrings;
  WKBLineString     WKBLineStrings[num_wkbLineStrings];
};
```

```
wkbMultiPolygon {
  byte              byteOrder;
  uint32            wkbType;     // 6=wkbMultiPolygon
  uint32            num_wkbPolygons;
  WKBPolygon        wkbPolygons[num_wkbPolygons];
};

WKBGeometry  {
  union {
    WKBPoint                point;
    WKBLineString           linestring;
    WKBPolygon              polygon;
    WKBMultiPoint           mpoint;
    WKBMultiLineString      mlinestring;
    WKBMultiPolygon         mpolygon;
  }
};
```

A figura a seguir mostra um exemplo de uma geometria na representação binária reconhecida utilizando a codificação NDR.

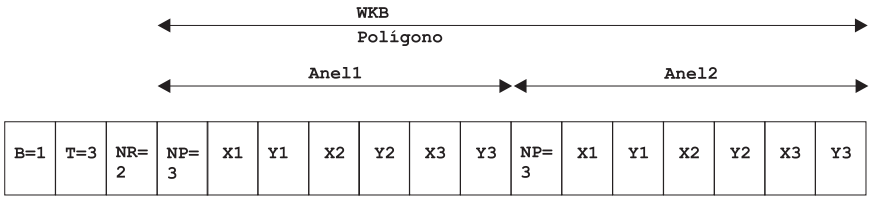


Figura 58. Representação Geométrica em Formato NDR.. (B=1) de tipo polígono (T=3) com 2 lineares (NR=2), em que cada anel tem 3 pontos (NP=3).

Referência Relacionada:

- "Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados" na página 299

Representação de Formatos

A representação de formatos é um padrão da indústria amplamente utilizado, definido por ESRI. Para obter uma descrição completa da representação de formatos, consulte o site do ESRI na Web em http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.

O tópico "Funções espaciais que convertem geometrias em e a partir de formatos de troca de dados" na seção do link relacionado abaixo explica as funções espaciais que aceitam e geram formato de dados de formatos.

Referência Relacionada:

- "Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados" na página 299

Representação GML (Geography Markup Language)

O DB2 Spatial Extender possui várias funções que geram geometrias a partir de representações em representação de GML (Geography Markup Language).

Consulte "Funções espaciais que convertem geometrias em e a partir de formatos de troca de dados" na seção do link relacionado abaixo para obter uma descrição

detalhada das funções fornecidas pelo DB2 Spatial Extender que convertem valores de geometrias em e a partir de representação GML.

A GML (Geography Markup Language) é uma codificação XML para informações geográficas definidas pela especificação OpenGIS Consortium ″Geography Markup Language V2″. Esta especificação do OpenGIS Consortium pode ser encontrada em http://www.opengis.org/techno/implementation.htm.

Referência Relacionada:

- “Funções Espaciais que Convertem Valores de Geometria em Formatos de Troca de Dados” na página 299

Capítulo 26. Sistemas de Coordenadas Suportados

Este capítulo fornece informações de referência sobre os valores de coordenadas utilizados para interpretar dados espaciais. São abordados os seguintes tópicos:

- Visão geral de sistemas de coordenadas
- Unidades lineares suportadas
- Unidades angulares suportadas
- Esferóides suportados
- Dados geodéticos suportados
- Meridianos principais suportados
- Projeções do mapa suportadas

Sistemas de Coordenadas Suportados

Este tópico fornece uma explicação da sintaxe de sistemas de coordenadas e lista os valores de sistemas de coordenadas que são suportados pelo DB2 Spatial Extender.

Sintaxe de Sistemas de Coordenadas

A representação de texto reconhecida dos sistemas de referência espacial fornece uma representação textual padrão para informações do sistema de coordenadas. As definições da representação de texto bem conhecido são definidas pela especificação OGC ″Simple Features for SQL″ e pelo ISO SQL/MM Part 3: Spatial standard.

Um sistema de coordenadas é um sistema de coordenadas geográficas (latitude-longitude), projetadas (X,Y) ou geocêntricas (X,Y,Z). O sistema de coordenadas é composto de vários objetos. Cada objeto tem uma palavra-chave em maiúsculas (por exemplo, DATUM ou UNIT) seguido dos parâmetros de definição, delimitados por vírgulas, do objeto entre colchetes. Alguns objetos são compostos de outros objetos, portanto, o resultado é uma estrutura aninhada.

Nota: As implementações estão livres para substituir os colchetes padrão ( ) por chaves [ ] e devem poder ler os dois formatos de colchetes.

A definição EBNF (Extended Backus Naur Form) da representação de cadeia de um sistema de coordenadas que utiliza colchetes é a seguinte (consulte a nota acima referente à utilização de colchetes):

```
<sistema_de_coordenadas> = <cs_projetada> |
<cs_geográfico> | <cs_geocêntricas>
<cs_projetada> = PROJCS["<nome>",
<cs_geográfico>, <projeção>, {<parâmetro>,}*
<unidade linear>]
<projeção> = PROJECTION["<nome>"]
<parâmetro> = PARAMETER["<nome>",
<valor>]

<valor> = <número>
```

O tipo de sistema de coordenadas é identificado pela palavra-chave utilizada:

PROJCS
Um sistema de coordenadas do conjunto de dados será identificado pela palavra-chave PROJCS se os dados estiverem nas coordenadas projetadas

GEOGCS
Um sistema de coordenadas do conjunto de dados será identificado pela palavra-chave GEOGCS se os dados estiverem nas coordenadas geográficas

GEOCCS
Um sistema de coordenadas do conjunto de dados será identificado pela palavra-chave GEOGCS se os dados estiverem nas coordenadas geocêntricas

A palavra-chave PROJCS é seguida de todas as partes" que definem o sistema de coordenadas projetadas. A primeira parte de qualquer objeto é sempre o nome. Vários objetos seguem o nome do sistema de coordenadas projetadas: o sistema de coordenadas geográficas, a projeção de mapas, um ou mais parâmetros e a unidade linear de medida. Todos os sistemas de coordenadas projetadas são baseados num sistema de coordenadas geográficas, portanto, esta sessão descreve primeiro as partes específicas de um sistema de coordenadas projetadas. Por exemplo, a zona UTM 10N nos dados NAD83 está definida:

```
PROJCS["NAD_1983_UTM_Zone_10N",
<cs_geográfico>,
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]]
```

O nome e vários objetos definem o objeto do sistema de coordenadas geográficas em turnos: o dado, o meridiano principal e a unidade angular de medida.

```
<cs_geográfico> = GEOGCS["<nome>", <datum>,
<meridiano_principal>, <unidade_angular>]
<datum> = DATUM["<nome>", <esferóide>]
<esferóide> = SPHEROID["<nome>", <semi-eixo_maior>, <condensação_inversa>]
<semi-eixo_maior> = <número>
<condensação_inversa> = <número>
<meridiano_principal> = PRIMEM["<nome>", <longitude>]
<longitude> = <número>
```

O semi-eixo maior é medido em metros e deve ser maior do que zero.

A cadeia do sistema de coordenadas geográficas para a zona UTM 10 em NAD83:

```
GEOGCS["GCS_North_American_1983",
DATUM["D_Norte_americano_1983",
SPHEROID["GRS_1980",6378137,298.257222101]],
PRIMEM["Greenwich",0],
UNIT["Degree",0.0174532925199433]]
```

O objeto UNIT pode representar unidade angular ou linear de medidas:

```
<unidade_angular> = <unidade>
<unidade_linear> = <unidade>
<unidade> = UNIT["<nome>", <fator_de_conversão>]
<fator_de_conversão> = <número>
```

O fator de conversão especifica o número de metros (para uma unidade linear) ou o número de radianos (para uma unidade angular) por unidade e deve ser maior que zero.

Assim, a representação completa da cadeia da Zona UTM 10N é a seguinte:

```
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]]
```

Um sistema de coordenadas geométricas é semelhante a um sistema de coordenadas geográficas:

```
<cs_geocêntrica> = GEOCCS["<nome>", <datum>, <meridiano_principal>,
 <unidade_linear>]
```

Unidades Lineares Suportadas

Tabela 58. Unidades Lineares Suportadas

Unidade	Fator de Conversão
Metro	1,0
Pé (Internacional)	0,3048
Pé Americano	12/39,37
Pé Americano Modificado	12,0004584/39,37
Pé de Clarke	12/39,370432
Pé Indiano	12/39,370141
Ligação	7.92/39.370432
Link (Benoit)	7.92/39.370113
Link (Sears)	7.92/39.370147
Cadeia (Benoit)	792/39.370113
Cadeia (Sears)	792/39.370147
Jarda (Índia)	36/39.370141
Jarda (Sears)	36/39.370147
Braça	1.8288
Milha Náutica	1852.0

Unidades Angulares Suportadas

Tabela 59. Unidades Angulares Suportadas

Unidade	Intervalo Válido para Latitude	Intervalo Válido para Longitude	Fator de Conversão
Radiano	–pi/2 e pi/2 radianos (inclusive)	–pi e pi radianos (inclusive)	1,0
Grau Decimal	–90 e 90 graus (inclusive)	–180 e 180 graus (inclusive)	pi/180
Minuto Decimal	–5400 e 5400 minutos (inclusive)	–10800 e 10800 minutos (inclusive)	(pi/180)/60
Segundo Decimal	–324000 e 324000 segundos (inclusive)	–648000 e 648000 segundos (inclusive)	(pi/180)*3600
Gon	–100 e 100 gradianos (inclusive)	–200 e 200 gradianos (inclusive)	pi/200

Tabela 59. Unidades Angulares Suportadas (continuação)

Unidade	Intervalo Válido para Latitude	Intervalo Válido para Longitude	Fator de Conversão
Grade	–100 e 100 gradianos (inclusive)	–200 e 200 gradianos (inclusive)	pi/200

Esferóides Suportados

Tabela 60. Esferóides Suportados

Nome	Eixo Semi-principal	Condensação Inversa
Airy 1830	6377563.396	299.3249646
Airy Modificado 1849	6377340.189	299.3249646
Average Terrestrial System 1977	6378135.0	298.257
Australian National Spheroid	6378160.0	298.25
Bessel 1841	6377397.155	299.1528128
Bessel Modificado	6377492.018	299.1528128
Bessel Namibia	6377483.865	299.1528128
Clarke 1858	6378293.639	294.260676369
Clarke 1866	6378206.4	294.9786982
Clarke 1866 (Michigan)	6378450.047	294.978684677
Clarke 1880	6378249.138	293.466307656
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
Clarke 1880 (SGA 1922)	6378249.2	293.46598
Everest (Definição de 1830)	6377299.36	300.8017
Everest 1830 Modified	6377304.063	300.8017
Everest Adjustment 1937	6377276.345	300.8017
Everest 1830 (Definição de 1962)	6377301.243	300.8017255
Everest 1830 (Definição de 1967)	6377298.556	300.8017
Everest 1830 (Definição de 1975)	6377299.151	300.8017255
Everest 1969 Modificado	6377295.664	300.8017
Fischer 1960	6378166.0	298.3
Fischer 1968	6378150 .0	298.3
Fischer Modificado	6378155 .0	298.3
GEM 10C	6378137.0	298.257222101
GRS 1967	6378160.0	298.247167427
GRS 1967 Truncado	6378160.0	298.25
GRS 1980	6378137.0	298.257222101

Tabela 60. Esferóides Suportados (continuação)

Nome	Eixo Semi-principal	Condensação Inversa
Helmert 1906	6378200.0	298.3
Hough 1960	6378270.0	297.0
Indonesian National Spheroid	6378160.0	298.247
Internacional 1924	6378388.0	297.0
International 1967	6378160.0	298.25
Krassowsky 1940	6378245.0	298.3
NWL 9D	6378145.0	298.25
NWL 10D	6378135.0	298.26
OSU 86F	6378136.2	298.25722
OSU 91A	6378136.3	298.25722
Plessis 1817	6376523.0	308.64
Sphere	6371000.0	0.0
Sphere (ArcInfo)	6370997.0	0.0
Struve 1860	6378298.3	294.73
Walbeck	6376896.0	302.78
War Office	6378300.0	296.0
WGS 1966	6378145.0	298.25
WGS 1972	6378135.0	298.26
WGS 1984	6378137.0	298.257223563

Dados Geodéticos Suportados

Tabela 61. Dados Geodéticos Suportados

Nome	Datum Geodésico
Adindan	Lisboa
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

Tabela 61. Dados Geodéticos Suportados (continuação)

Nome	Datum Geodésico
Reseau National Belge 1972	North American Datum 1927
Bermuda 1957	North American Datum 1983
Bern 1898	Nahrwan 1967
Bern 1938	Naparima 1972
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
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
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	Palestina 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

Tabela 61. Dados Geodéticos Suportados (continuação)

Nome	Datum Geodésico
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	Estocolmo 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
Kuwait Oil Company	WGS 1972 Transit Broadcast Ephemeris
La Canoa	WGS 1984
Lake	Yacare
Leigon	Yoff
Liberia 1964	Zanderij

Meridianos Principais Suportados

Tabela 62. Meridianos Principais Suportados

Localização	Coordenadas
Greenwich	0° 0' 0"
Bern	7° 26' 22.5" E
Bogota	74° 4' 51.3" W
Bruxelas	4° 22' 4.71" E
Ferro	17° 40' 0" W
Jakarta	106° 48' 27.79" E
Lisboa	9° 7' 54.862" W
Madri	3° 41' 16.58" W
Paris	2° 20' 14.025" E
Roma	12° 27' 8.4" E
Estocolmo	18° 3' 29" E

Projeções de Mapas Suportadas

Tabela 63. Projeções Cilíndricas

Projeções Cilíndricas	Projeções Pseudocilíndricas
Behrmann	Craster parabolic
Cassini	Eckert I
Cylindrical equal area	Eckert II
Equiretangular	Eckert III
Gall's stereographic	Eckert IV
Gauss-Kruger	Eckert V
Mercator	Eckert VI
Miller cylindrical	McBryde-Thomas flat polar quartic
Oblique	Mercator (Hotine) Mollweide
Plate-Carée	Robinson
Times	Sinusoidal (Sansom-Flamsteed)
Transverse Mercator	Winkel I

Tabela 64. Projeções Cônicas

Nome	Projeção Cônica
Albers conic equal-area	Chamberlin trimetric
Bipolar oblique conformal conic	Two-point equidistant
Bonne	Hammer-Aitoff equal-area
Equidistant conic	Van der Grinten I
Lambert conformal conic	Miscellaneous
Polyconic	Alaska series E
Simple conic	Alaska Grid (Modified-Stereographic by Snyder)

Tabela 65. Parâmetros de Projeção do Mapa

Parâmetro	Descrição
central_meridian	A linha da longitude escolhida como a origem das coordenadas x.
scale_factor	Scale_factor geralmente é utilizado para reduzir a quantidade de distorção em uma projeção de mapa.
standard_parallel_1	Uma linha de latitude que geralmente não apresenta distorção. Usada também para "latitude de escala verdadeira."
standard_parallel_2	Uma linha de longitude que geralmente não apresenta distorção.
longitude_of_center	A longitude que define o ponto central da projeção do mapa.
latitude_of_center	A latitude que define o ponto central da projeção do mapa.
longitude_of_origin	A longitude escolhida como a origem das coordenadas x.

Tabela 65. Parâmetros de Projeção do Mapa (continuação)

Parâmetro	Descrição
latitude_of_origin	A latitude escolhida como a origem das coordenadas y.
false_easting	Um valor incluído em coordenadas x para que todos os valores de coordenadas x sejam positivos.
false_northing	Um valor incluído em coordenadas y para que todas as coordenadas y sejam positivas.
azimuth	O ângulo leste do norte que define a linha central de uma projeção oblíqua.
longitude_of_point_1	A longitude do primeiro ponto necessário para uma projeção de mapa.
latitude_of_point_1	A latitude do primeiro ponto necessário para uma projeção de mapa.
longitude_of_point_2	A longitude do segundo ponto necessário para uma projeção de mapa.
latitude_of_point_2	A latitude do segundo ponto necessário para uma projeção de mapa.
longitude_of_point_3	A longitude do terceiro ponto necessário para uma projeção de mapa.
latitude_of_point_3	A latitude do terceiro ponto necessário para uma projeção de mapa.
landsat_number	O número de um satélite Landsat.
path_number	O número de caminho orbital de um determinado satélite.
perspective_point_height	O peso acima da terra do ponto de perspectiva da projeção do mapa.
fipszone	Número de zona do Sistema de Coordenadas Planas do Estado.
zone	Número de zona UTM.

Material de referência

Apêndice A. Procedimentos Armazenados Reprovados

Este tópico descreve os procedimentos armazenados reprovados.

Nota: Recomendação: escreva todos os novos aplicativos utilizando os procedimentos armazenados definidos no DB2 Spatial Extender Versão 8 e atualize os aplicativos atuais para utilizar os procedimentos armazenados definidos na Versão 8.

Os procedimentos armazenados reprovados executam as tarefas resumidas na tabela abaixo.

Tabela 66. Procedimentos Armazenados Reprovados

Nome do Procedimento Armazenado	Tarefa do Procedimento Armazenado
db2gse.gse_enable_autogc	Ativação de um geocoder para manter automaticamente colunas espaciais sincronizadas com suas colunas de atributos correspondentes
db2gse.gse_enable_db	Ativação de um banco de dados para suportar operações espaciais
db2gse.gse_enable_idx	Criação de um índice para uma coluna espacial
db2gse.gse_enable_sref	Criação de um sistema de referência espacial
db2gse.gse_export_shape	Exportação de uma camada e de sua tabela associada para um arquivo modelo
db2gse.gse_disable_autogc	Desativação de um geocoder para que ele não possa manter automaticamente colunas espaciais sincronizadas com suas colunas de atributos correspondentes
db2gse.gse_disable_db	Desativação de suporte para operações espaciais em um banco de dados
db2gse.gse_disable_sref	Eliminação de um sistema de referência espacial
db2gse.gse_import_shape	Importação de uma camada e de sua tabela associada de um arquivo de transferência ESRI_SDE
db2gse.gse_register_gc	Registro de um geocoder diferente do padrão
db2gse.gse_register_layer	Registro de uma coluna espacial como uma camada
db2gse.gse_run_gc	Executar um geocoder no modo batch
db2gse.gse_unregist_gc	Cancelamento de registro de um geocoder diferente do padrão
db2gse.gse_unregist_layer	Cancelamento de registro de uma camada

db2gse.gse_enable_autogc

Use este procedimento armazenado para:

- Criar disparos que mantêm uma coluna espacial sincronizada com sua coluna ou colunas de atributos associados. Sempre que forem inseridos ou atualizados valores na coluna ou colunas de atributos, um disparo chamará um geocoder registrado para executar geocode dos valores inseridos ou atualizados e colocar os dados resultantes na coluna espacial.
- Reativar os disparos depois de serem temporariamente desativados.
- Estabelecer qual função será utilizada para executar geocode dos valores inseridos ou atualizados.

Autorização:

O ID do usuário com o qual este procedimento armazenado é chamado deve ter as seguintes autoridades ou privilégios:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual estão definidos os disparadores criados pelo procedimento armazenado.
- O privilégio CONTROL nesta tabela.
- Os privilégios ALTER, SELECT e UPDATE nesta tabela.

Parâmetros:

Tabela 67. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_enable_autogc.

Nome	Tipo de Dados	Descrição
operMode	SMALLINT	Valor que indica se os disparadores que iniciam a geocodificação serão criados pela primeira vez ou serão reativados após a desativação temporária. Este parâmetro não pode ser nulo. **Comentário:** Para criar os disparadores, use a macro GSE_AUTOGC_CREATE. Para reativá-los, use a macro GSE_AUTOGC_RECREATE. Para descobrir quais valores estão associados a estas macros, consulte o arquivo db2gse.h. No AIX, este arquivo está armazenado no diretório $DB2INSTANCE/sqllib/include/. No Windows NT, está armazenado no diretório %DB2PATH%\include\. Se o parâmetro operMode estiver definido como GSE_AUTOGC_CREATE, você deverá atribuir um identificador de um geocoder registrado ao parâmetro gcId.
layerSchema	VARCHAR(30)	Nome do esquema ao qual pertence a tabela especificada no parâmetro layerTable. Este parâmetro pode ser nulo. Se não for fornecido um valor para o parâmetro layerSchema, será assumido como padrão o ID do usuário com o qual o procedimento armazenado db2gse.gse_enable_autogc é chamado.
layerTable	VARCHAR(128)	Nome da tabela em que serão operados os disparadores criados ou reativados por este procedimento armazenado. Este parâmetro não pode ser nulo.

Tabela 67. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_enable_autogc. (continuação)

Nome	Tipo de Dados	Descrição
layerColumn	VARCHAR(128)	Nome da coluna espacial que deverá ser mantida pelos disparadores que este procedimento armazenado cria ou reativa. Este parâmetro não pode ser nulo. O parâmetro layerColumn deve fazer referência a uma coluna que tenha sido registrada como camada da tabela.
gcId	INTEGER	Identificador do geocoder que será chamado pelos disparadores de inserção e atualização que este procedimento armazenado cria ou reativa. Este parâmetro não pode ser nulo se o parâmetro operMode for definido como GSE_AUTOGC_CREATE. Ele poderá ser nulo se operMode estiver definido como GSE_AUTOGC_RECREATE.
precisionLevel	INTEGER	O grau em que os dados fonte devem coincidir dados de referência correspondentes para que o geocoder processe os dados fonte com êxito. Este parâmetro não pode ser nulo se o parâmetro operMode for definido como GSE_AUTOGC_CREATE. Ele poderá ser nulo se operMode estiver definido como GSE_AUTOGC_RECREATE. O nível de precisão pode variar de 1 a 100 por cento.
vendorSpecific	VARCHAR(256)	Informações técnicas fornecidas pelo fornecedor; por exemplo, o caminho e nome de um arquivo que o fornecedor usa para definir parâmetros. Este parâmetro não pode ser nulo se o parâmetro operMode for definido como GSE_AUTOGC_CREATE. Ele poderá ser nulo se operMode estiver definido como GSE_AUTOGC_RECREATE.

Resultados:

Tabela 68. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_enable_autogc.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro completa, conforme construída no servidor.

db2gse.gse_enable_db

Use este procedimento armazenado para fornecer a um banco de dados os recursos de que ele precisa para armazenar dados espaciais e suportar operações. Estes recursos incluem tipos de dados espaciais, um tipo de índice espacial, tabelas e exibições do catálogo, funções fornecidas e outros procedimentos armazenados. A biblioteca externa e o nome da função para este procedimento armazenado é db2gse.gse_enable_db.

Autorização:

O ID de usuário com qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM no banco de dados que está sendo ativado.

Resultados:

Tabela 69. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_enable_db.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_enable_idx

Use este procedimento armazenado para criar um índice para uma coluna espacial.

Autorização:

O ID de usuário com o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual o índice ativado deverá ser usado.
- O privilégio CONTROL ou INDEX nesta tabela.

Parâmetros:

Tabela 70. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_enable_idx.

Nome	Tipo de Dados	Descrição
layerSchema	VARCHAR(30)	Nome do esquema ao qual pertence a tabela especificada no parâmetro layerTable. Este parâmetro pode ser nulo. É necessário fornecer um valor para este parâmetro. O parâmetro pode ser um valor NULL.
layerTable	VARCHAR(128)	Nome da tabela na qual deverá ser definido o índice que você está criando. Este parâmetro não pode ser nulo.

Tabela 70. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_enable_idx. (continuação)

Nome	Tipo de Dados	Descrição
layerColumn	VARCHAR(128)	Nome da coluna ativada espacialmente que deverá ser pesquisada com a ajuda do índice que está sendo criado. Este parâmetro não pode ser nulo.
indexName	VARCHAR(128)	Nome do índice que deverá ser criado. Este parâmetro não pode ser nulo. Não especifique um nome de esquema. O DB2 Spatial Extender atribui automaticamente o índice ao esquema referido pelo parâmetro layerSchema.
gridSize1	DOUBLE	Número que indica qual deve ser a granulosidade da melhor grade de índice. Este parâmetro não pode ser nulo.
gridSize2	DOUBLE	Número que indica (1) que não deve haver segunda grade para este índice ou (2) qual deve ser a granulosidade da segunda grade. Este parâmetro pode ser nulo. Se a segunda grade não deve existir, especifique 0. Se desejar uma segunda grade, ela deve ser menos granular do que a grade indicada por gridSize1.
gridSize3	DOUBLE	Número que indica (1) que não deve haver terceira grade para este índice ou (2) qual deve ser a granulosidade da terceira grade. Este parâmetro pode ser nulo. Se a terceira grade não deve existir, especifique 0. Se desejar uma terceira grade, ela deve ser menos granular do que a grade indicada por gridSize2.

Resultados:

Tabela 71. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_enable_idx.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_enable_sref

Utilize este procedimento armazenado para especificar como os números negativos e decimais em um sistema de coordenadas específico devem ser convertidos em inteiros positivos para que o DB2 Spatial Extender possa armazená-los. Suas especificações são denominadas coletivamente *sistema de referência espacial*. Quando

este procedimento armazenado estiver processado, as informações sobre o sistema de referência espacial serão incluídas na exibição do catálogo DB2GSE.SPATIAL_REF_SYS.

Autorização:

Nenhuma é necessária.

Parâmetros:

Tabela 72. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_enable_sref.

Nome	Tipo de Dados	Descrição
srId	INTEGER	Um identificador numérico para o sistema de referência espacial. Este identificador deve ser exclusivo no banco de dados ativado espacialmente. Este parâmetro não pode ser nulo.
srName	VARCHAR(64)	Descrição breve do sistema de referência espacial. Esta descrição deve ser exclusiva no banco de dados ativado espacialmente. Este parâmetro não pode ser nulo.
falsex	DOUBLE	Um número que, quando subtraído de um valor negativo da coordenada X, deixa um número não-negativo (ou seja, um número positivo ou zero). Este parâmetro não pode ser nulo.
falsey	DOUBLE	Um número que, quando subtraído de um valor negativo da coordenada Y, permite um número não-negativo (ou seja, um número positivo ou zero). Este parâmetro não pode ser nulo.
xyunits	DOUBLE	Um número que, quando multiplicado por uma coordenada X decimal ou uma coordenada Y decimal, produz um inteiro que pode ser armazenado como um item de dados de 32 bits. Este parâmetro não pode ser nulo.
falsez	DOUBLE	Um número que, quando subtraído de um valor negativo da coordenada Z, permite um número não-negativo (ou seja, um número positivo ou zero). Este parâmetro não pode ser nulo.
zunits	DOUBLE	Um número que, quando multiplicado por uma coordenada Z decimal, produz um inteiro que pode ser armazenado como um item de dados de 32 bits. Este parâmetro não pode ser nulo.
falsem	DOUBLE	Um número que, quando subtraído de uma medida negativa, permite um número não-negativo (ou seja, um número positivo ou zero). Este parâmetro não pode ser nulo.

Tabela 72. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_enable_sref. (continuação)

Nome	Tipo de Dados	Descrição
munits	DOUBLE	Um número que, quando multiplicado por uma medida decimal, produz um inteiro que pode ser armazenado como um item de dados de 32 bits. Este parâmetro não pode ser nulo.
scId	INTEGER	Identificador numérico do sistema de coordenadas do qual é derivado o sistema de referência espacial. Para saber o que é um identificador numérico de um sistema de coordenadas, consulte a exibição do catálogo DB2GSE.COORD_REF_SYS. Este parâmetro não pode ser nulo.

Resultados:

Tabela 73. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_enable_sref.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_export_shape

Use este procedimento armazenado para exportar uma camada e sua tabela associada a um arquivo de formato ou criar um novo arquivo de formato e exportar uma camada e sua tabela associada para este novo arquivo.

Autorização:

O ID de usuário com qual este procedimento armazenado é chamado deve ter o privilégio SELECT na tabela que deverá ser exportada.

Parâmetros:

Tabela 74. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_export_shape.

Nome	Tipo de Dados	Descrição
layerSchema	VARCHAR(30)	Nome do esquema ao qual pertence a tabela especificada no parâmetro layerTable. Este parâmetro pode ser nulo. Se não for fornecido um valor para o parâmetro layerSchema, será assumido como padrão o ID do usuário com o qual o procedimento armazenado db2gse.gse_export_shape é chamado.
layerTable	VARCHAR(128)	Nome da tabela que você está exportando. Este parâmetro não pode ser nulo.

Tabela 74. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_export_shape. (continuação)

Nome	Tipo de Dados	Descrição
layerColumn	VARCHAR(30)	Nome da coluna que foi registrada como sendo a camada que você está exportando. Este parâmetro não pode ser nulo.
fileName	VARCHAR(128)	Nome do arquivo de formato para o qual a camada especificada deverá ser exportada. Este parâmetro não pode ser nulo.
whereClause	VARCHAR(1024)	O corpo de whereClause. Define uma restrição em um conjunto de linhas a serem exportadas. A cláusula pode referenciar qualquer coluna de atributo na tabela que você está exportando. A palavra-chave WHERE não é necessária nesta cláusula. Este parâmetro pode ser nulo.

Resultados:

Tabela 75. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_export_shape.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

Restrição:

É possível exportar somente uma camada por vez.

db2gse.gse_disable_autogc

Use este procedimento armazenado para eliminar ou desativar temporariamente disparadores que mantêm uma coluna espacial sincronizada com suas colunas de atributos. Por exemplo, é aconselhável desativar os disparadores enquanto você geocodifica os valores na coluna ou colunas atributos no modo batch.

Autorização:

O ID do usuário com o qual este procedimento armazenado é chamado deve ter uma das seguintes autoridades, privilégios ou conjunto de privilégios:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual estão definidos os disparos que estão sendo eliminados ou desativados temporariamente.
- O privilégio CONTROL nesta tabela.
- Os privilégios ALTER e UPDATE nesta tabela.

Nota: Para os privilégios CONTROL e ALTER, você deve ter autoridade DROPIN no esquema DB2GSE.

Parâmetros:

Tabela 76. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_disable_autogc.

Nome	Tipo de Dados	Descrição
operMode	SMALLINT	Indica se os disparadores deverão ser eliminados ou desativados temporariamente. Os disparos eliminados não possuem efeito sobre as instruções SQL. Os disparos desativados temporariamente podem ser recriados sem precisar especificar novamente os parâmetros definidos anteriormente. Este parâmetro não pode ser nulo. Para eliminar disparos, utilize a macro GSE_AUTOGC_DROP. Para desativá-los temporariamente, use a macro GSE_AUTOGC_INVALIDATE. Para descobrir quais valores estão associados a estas macros, consulte o arquivo db2gse.h. No AIX, este arquivo está armazenado no diretório $DB2INSTANCE/sqllib/include/. No Windows NT, está armazenado no diretório %DB2PATH%\include\.
layerSchema	VARCHAR(30)	Nome do esquema ao qual pertence a tabela ou exibição especificados no parâmetro layerTable. Este parâmetro pode ser nulo. Se não for fornecido um valor para o parâmetro layerSchema, será assumido como padrão o ID do usuário com o qual o procedimento armazenado db2gse.gse_disable_autogc é chamado.
layerTable	VARCHAR(128)	Nome da tabela na qual estão definidos os disparos que você deseja eliminar ou desativar temporariamente. Este parâmetro não pode ser nulo.
layerColumn	VARCHAR(128)	Nome da coluna ativada espacialmente que é mantida pelos disparos que você deseja eliminar ou desativar temporariamente. Este parâmetro não pode ser nulo.

Resultados:

Tabela 77. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_disable_autogc.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.

Tabela 77. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_disable_autogc. (continuação)

Nome	Tipo de Dados	Descrição
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_disable_db

Utilize este procedimento armazenado para remover recursos que permitem ao DB2 Spatial Extender armazenar dados espaciais e suportar operações que são executadas nesses dados.

O objetivo deste procedimento armazenado é ajudá-lo a resolver problemas ou questões que surgirem após você ativar o banco de dados para operações espaciais, porém *antes* inclua qualquer coluna ou dados da tabela nele. Por exemplo, se, depois de ativar um banco de dados para operações espaciais, você decidir utilizar o DB2 Spatial Extender para outro banco de dados. Contanto que não tenha definido colunas espaciais ou importado dados espaciais, você pode chamar este procedimento armazenado para remover todos os recursos espaciais do primeiro banco de dados.

Autorização:

O ID de usuário sob o qual este procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM no banco de dados a partir do qual os recursos do DB2 Spatial Extender devem ser removidos.

Resultados:

Tabela 78. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_disable_db.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_disable_sref

Use este procedimento armazenado para eliminar um sistema de referência espacial. Quando este procedimento armazenado estiver processado, as informações sobre o sistema de referência espacial serão removidas da exibição do catálogo DB2GSE.SPATIAL_REF_SYS.

Pré-requisitos:

Antes de eliminar um sistema de referência espacial, você deverá cancelar o registro de qualquer camada que o utilize. Se tais camadas permanecerem sem registro, a solicitação de eliminação do sistema de referência espacial será rejeitada.

Autorização:

Nenhuma é necessária.

Processo:

Tabela 79. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_disable_sref.

Nome	Tipo de Dados	Descrição
srId	INTEGER	Identificador numérico do sistema de referência espacial que será eliminado. Este parâmetro não pode ser nulo.

Resultados:

Tabela 80. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_disable_sref.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_import_shape

Utilize este procedimento armazenado para importar um arquivo modelo ESRI para um banco de dados que é ativado para operações espaciais. O procedimento armazenado pode operar de uma destas duas formas:

- Se o arquivo modelo destinar-se a uma tabela existente que tenha uma coluna de camada registrada, o DB2 Spatial Extender carregará a tabela com os dados do arquivo.
- Se o arquivo modelo destinar-se a uma tabela que não existe, o DB2 Spatial Extender criará uma tabela que tem uma coluna espacial, registrará esta coluna como uma camada e carregará a camada e as outras colunas da tabela com os dados do arquivo.

Ao importar um conjunto de representações do formato ESRI, você recebe, pelo menos, dois arquivos. Todos os arquivos possuem o mesmo prefixo para o nome, mas extensões diferentes. Por exemplo, as extensões dos dois arquivos que você sempre recebe são .shp e .shx.

Para receber os arquivos de um conjunto de representações de formato, atribua o nome que os arquivos possuem em comum ao parâmetro fileName. Não especifique uma extensão. Dessa forma, você pode assegurar que todos os arquivos necessários — o arquivo .shp, o arquivo .shx e quaisquer outros que possam ser incluídos — serão importados.

Por exemplo, suponha que um conjunto de representações de formatos ESRI esteja armazenado nos arquivos chamados Lakes.shp e Lakes.shx. Ao importar estas representações, você deve atribuir somente o nome Lakes no parâmetro fileName.

Os arquivos de transferência SDE possuem nomes, mas não extensões. Portanto, ao importar um arquivo de transferência SDE, você atribui seu nome, sem a extensão, ao parâmetro fileName.

Autorização:

O ID de usuário com o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual serão carregados os dados de formato importados.
- O privilégio CONTROL nesta tabela.

Parâmetros:

Tabela 81. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_import_shape.

Nome	Tipo de Dados	Descrição
layerSchema	VARCHAR(30)	Nome do esquema ao qual pertence a tabela ou exibição especificados no parâmetro layerTable. Este parâmetro pode ser nulo. Se não for fornecido um valor para o parâmetro layerSchema, será assumido como padrão o ID do usuário com o qual o procedimento armazenado db2gse.gse_import_shape é chamado.
layerTable	VARCHAR(128)	Nome da tabela na qual deve ser carregado o arquivo de formato é importado. Este parâmetro não pode ser nulo.
layerColumn	VARCHAR(30)	Nome da coluna que foi registrada como camada na qual serão carregados os dados de formato. Este parâmetro não pode ser nulo.
fileName	VARCHAR(128)	Nome do arquivo de formato que deverá ser importado. Este parâmetro não pode ser nulo.
exceptionFile	VARCHAR(128)	Caminho e nome do arquivo no qual serão armazenados os formatos que não puderam ser importados. Este é um novo arquivo que será criado quando o procedimento armazenado db2gse.gse_import_shape for executado. Atribua um nome do arquivo, mas não uma extensão, ao parâmetro exceptionFile Este parâmetro não pode ser nulo.
srId	INTEGER	Identificador do sistema de referência espacial a ser usado pela camada na qual os dados de formato serão carregados. Este parâmetro pode ser nulo. Se este identificador não estiver especificado, a transformação interna será definida como a resolução máxima possível para o arquivo modelo.
commitScope	INTEGER	Número de registros por ponto de verificação. Este parâmetro que é I foi nulo.

Resultados:

Tabela 82. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_import_shape.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_register_gc

Use este procedimento armazenado para registrar um geocoder diferente do padrão. Para saber se um geocoder já está registrado, consulte a exibição do catálogo DB2GSE.SPATIAL_GEOCODER.

Autorização:

O ID de usuário com qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM no banco de dados que contém o geocoder que este procedimento armazenado registra.

Parâmetros:

Tabela 83. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_register_gc.

Nome	Tipo de Dados	Descrição
gcId	INTEGER	Identificador numérico do geocoder que você está registrando. Este identificador deve ser exclusivo no banco de dados. Este parâmetro não pode ser nulo.
gcName	VARCHAR(64)	Descrição breve do geocoder que você está registrando. Esta descrição deve ser uma cadeia de caracteres exclusiva no banco de dados. Este parâmetro não pode ser nulo.
vendorName	VARCHAR(64)	Nome do fornecedor que ofereceu o geocoder que você está registrando. Este parâmetro não pode ser nulo.
primaryUDF	VARCHAR(256)	Nome completo do geocoder que você está registrando. Este parâmetro não pode ser nulo.

Tabela 83. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_register_gc. (continuação)

Nome	Tipo de Dados	Descrição
precisionLevel	INTEGER	O grau em que os dados fonte devem coincidir dados de referência correspondentes para que o geocoder processe os dados fonte com êxito. O nível de precisão pode variar de 1 a 100 por cento. Este parâmetro não pode ser nulo.
vendorSpecific	VARCHAR(256)	Informações técnicas fornecidas pelo fornecedor; por exemplo, o caminho e nome de um arquivo que o fornecedor usa para definir parâmetros. Este parâmetro pode ser nulo.
geoArea	VARCHAR(256)	Área geográfica a ser geocodificada. Este parâmetro pode ser nulo.
description	VARCHAR(256)	Observações apresentadas pelo fornecedor Este parâmetro pode ser nulo.

Resultados:

Tabela 84. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_register_gc.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_register_layer

Use este procedimento armazenado para registrar uma coluna espacial como camada. Quando este procedimento armazenado estiver processado, as informações sobre a camada que está sendo registrada serão incluídas na exibição do catálogo DB2GSE.GEOMETRY_COLUMNS.

Autorização:

O ID de usuário com o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Para uma camada da tabela:
  - Autoridade SYSADM ou DBADM no banco de dados que contém a tabela à qual pertence esta camada.
  - O privilégio CONTROL ou ALTER nesta tabela.
- Para uma camada da exibição:
  - O privilégio SELECT na tabela base que contém (1) os dados de endereço que serão geocodificados para esta camada e (2) os dados espaciais que resultarem da geocodificação.

Parâmetros:

Tabela 85. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_register_layer.

Nome	Tipo de Dados	Descrição
layerSchema	INTEGER(30)	Nome do esquema ao qual pertence a tabela ou exibição especificados no parâmetro layerTable. Este parâmetro pode ser nulo. Se não for fornecido um valor para o parâmetro layerSchema, será assumido como padrão o ID do usuário com o qual o procedimento armazenado db2gse.gse_register_layer é chamado.
layerTable	VARCHAR(128)	Nome da tabela ou exibição que contém a coluna que está sendo registrada como camada. Este parâmetro não pode ser nulo.
layerColumn	VARCHAR(128)	Nome da coluna que está sendo registrada como camada. Para uma tabela, se a coluna não existir, o DB2 Spatial Extender a incluirá utilizando a instrução ALTER. Para uma exibição, a coluna já deve existir. Pode ser especificada apenas uma coluna para o parâmetro layerColumn. Portanto, quando você registra várias colunas de uma tabela ou exibição como camadas, é necessário executar este procedimento armazenado separadamente para cada coluna. Este parâmetro não pode ser nulo.
layerTypeName	VARCHAR(64)	Tipo de dados da coluna que está sendo registrada como camada. São aceitos apenas os tipos de dados fornecidos pelo DB2 Spatial Extender. Você deve especificar o tipo de dados em maiúsculas, por exemplo: `ST_POINT` Não é necessário especificar um nome de esquema, pois ele é incluído automaticamente. Este parâmetro não poderá ser nulo se a coluna for uma coluna de tabela que deve ser criada quando esse procedimento armazenado for processado. Caso contrário, se a coluna for uma coluna existente dentro de uma tabela ou exibição, esse parâmetro poderá ser nulo.
srId	INTEGER	Identificador do sistema de referência espacial usado para esta camada. Este parâmetro não pode ser nulo para a camada de uma tabela. O DB2 Spatial Extender ignora este parâmetro quando você registra uma camada de exibição.

Tabela 85. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_register_layer. (continuação)

Nome	Tipo de Dados	Descrição
geoSchema	VARCHAR(30)	O esquema da tabela que suporta a exibição à qual a coluna pertence. Este parâmetro se aplica quando você registra uma coluna de exibição como uma camada. Este parâmetro pode ser nulo quando você registra a coluna de uma exibição como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna da tabela como uma camada. Exibições baseadas em mais de uma tabela base ou outras exibições não são suportadas por este parâmetro. Se não for especificado um valor para o parâmetro geoSchema, será assumido como padrão o valor do parâmetro layerSchema.
geoTable	VARCHAR(128)	O nome da tabela que suporta a exibição à qual a coluna pertence. Este parâmetro se aplica quando você registra uma coluna de exibição como uma camada. Exibições baseadas em mais de uma tabela base ou outras exibições não são suportadas por este parâmetro. Este parâmetro pode ser nulo quando você registra a coluna de uma exibição como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna da tabela como uma camada.
geoColumn	VARCHAR(128)	O nome da coluna da tabela que suporta esta coluna de exibição. Este parâmetro se aplica quando você registra uma coluna de exibição como uma camada. Exibições baseadas em mais de uma tabela base ou outras exibições não são suportadas por este parâmetro. Este parâmetro pode ser nulo quando você registra a coluna de uma exibição como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna da tabela como uma camada.
nAttributes	SMALLINT	Número de colunas que contêm os dados fonte que deverão ser geocodificados para esta camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada.

Tabela 85. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_register_layer. (continuação)

Nome	Tipo de Dados	Descrição
attr1Name	VARCHAR(128)	Nome da primeira coluna que contém os dados fonte que deverão ser geocodificados para esta camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. Se você pretende usar o geocoder padrão, precisará armazenar os endereços das ruas na coluna attr1Name.
attr2Name	VARCHAR(128)	Nome da segunda coluna que contém os dados fonte que deverão ser geocodificados para essa camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. Se você pretende usar o geocoder padrão, precisará armazenar os nomes das cidades na coluna attr2Name.
attr3Name	VARCHAR(128)	Nome da terceira coluna que contém os dados fonte que devem ser geocodificados para essa camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. Se você pretende usar o geocoder padrão, precisará armazenar os nomes ou abreviações dos estados na coluna attr3Name.
attr4Name	VARCHAR(128)	Nome da quarta coluna que contém os dados fonte que devem ser geocodificados para essa camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. Se você pretende usar o geocoder padrão, precisará armazenar os CEPs na coluna attr4Name.
attr5Name	VARCHAR(128)	Nome da quinta coluna que contém os dados fonte que devem ser geocodificados para essa camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. O gecoder padrão ignora a coluna Attr5Name.

Tabela 85. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_register_layer. (continuação)

Nome	Tipo de Dados	Descrição
attr6Name	VARCHAR(128)	Nome da sexta coluna que contém os dados fonte que devem ser geocodificados para essa camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. O gecoder padrão ignora a coluna Attr6Name.
attr7Name	VARCHAR(128)	Nome da sétima coluna que contém os dados fonte que devem ser geocodificados para essa coluna. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. O gecoder padrão ignora a coluna Attr7Name.
attr8Name	VARCHAR(128)	Nome da oitava coluna que contém os dados fonte que devem ser geocodificados para essa camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. O gecoder padrão ignora a coluna Attr8Name.
attr9Name	VARCHAR(128)	Nome da nona coluna que contém os dados fonte que devem ser geocodificados para essa camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. O gecoder padrão ignora a coluna Attr9Name.
attr10Name	VARCHAR(128)	Nome da décima coluna que contém os dados fonte que devem ser geocodificados para essa camada. Este parâmetro pode ser nulo quando você registra a coluna de uma tabela como uma camada. O DB2 Spatial Extender ignora este parâmetro quando você registra uma coluna de exibição como uma camada. O gecoder padrão ignora a coluna Attr10Name.

Resultados:

Tabela 86. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_register_layer.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.

Tabela 86. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_register_layer. (continuação)

Nome	Tipo de Dados	Descrição
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

Restrições:

Este procedimento armazenado não funciona nos seguintes tipos de tabelas:

- A = Alias
- H = Tabela de Hierarquia
- N = Pseudônimo
- S = Tabela de Resumo
- U = Tabela Representada
- W = Exibição Representada

As seguintes restrições também se aplicam:

- Se você estiver registrando uma coluna da exibição como camada, este registro deverá estar baseado numa coluna de tabela que já esteja registrada como camada.
- Não mais que dez colunas de atributos podem conter os dados que deverão ser geocodificados para a camada que você está registrando.

db2gse.gse_run_gc

Use este procedimento armazenado para executar um geocoder no modo batch.

Autorização:

O ID de usuário com o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Autoridade SYSADM ou DBADM no banco de dados que contém a tabela na qual o geocoder especificado deve agir.
- O privilégio CONTROL ou UPDATE nesta tabela.

Parâmetros:

Tabela 87. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_run_gc.

Nome	Tipo de Dados	Descrição
layerSchema	VARCHAR(30)	Nome do esquema ao qual pertence a tabela ou exibição especificados no parâmetro layerTable. Este parâmetro pode ser nulo. Se não for fornecido um valor para o parâmetro layerSchema, será assumido como padrão o ID do usuário com o qual db2gse.gse_run_gc é chamado.
layerTable	VARCHAR(128)	Nome da tabela que contém a coluna na qual os dados geocodificados serão inseridos. Este parâmetro não pode ser nulo.

Tabela 87. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_run_gc. (continuação)

Nome	Tipo de Dados	Descrição
layerColumn	VARCHAR(128)	Nome da coluna na qual os dados geocodificados serão inseridos. Este parâmetro não pode ser nulo.
gcId	INTEGER	Identificador do geocoder que você deseja executar. Este parâmetro pode ser nulo. Para localizar os identificadores de geocoders registrados, consulte a exibição do catálogo DB2GSE.SPATIAL_GEOCODER.
precisionLevel	INTEGER	O grau em que os dados fonte devem coincidir dados de referência correspondentes para que o geocoder processe os dados fonte com êxito. Este parâmetro pode ser nulo. O nível de precisão pode variar de 1 a 100 por cento.
vendorSpecific	VARCHAR(256)	Informações técnicas fornecidas pelo fornecedor; por exemplo, o caminho e nome de um arquivo que o fornecedor usa para definir parâmetros. Este parâmetro pode ser nulo.
whereClause	VARCHAR(256)	O corpo da cláusula WHERE. Define uma restrição ao conjunto de registros que serão geocodificados. A cláusula pode referenciar qualquer coluna de atributo na tabela em que o geocoder será operado. Este parâmetro pode ser nulo.
commitScope	INTEGER	Número de registros por ponto de verificação. Este parâmetro pode ser nulo.

Resultados:

Tabela 88. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_run_gc.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_unregist_gc

Use este procedimento armazenado para cancelar um registro diferente do geocoder padrão. Para encontrar informações sobre o geocoder do qual deseja cancelar o registro, consulte a exibição do catálogo DB2GSE.SPATIAL_GEOCODER.

Autorização:

O ID de usuário com o qual o procedimento armazenado é chamado deve ter autoridade SYSADM ou DBADM no banco de dados que contém o geocoder que deverá ter o registro cancelado.

Parâmetros::

Tabela 89. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_unregist_gc.

Nome	Tipo de Dados	Descrição
gcId	INTEGER	O identificador do geocoder que deverá ter o registro cancelado. Este parâmetro não pode ser nulo.

Resultados:

Tabela 90. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_unregist_gc.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

db2gse.gse_unregist_layer

Use este procedimento armazenado para cancelar o registro de uma camada. O procedimento armazenado faz isto:

- Removendo a definição da camada das tabelas do catálogo do DB2 Spatial Extender.
- Excluindo a restrição de verificação que o DB2 Spatial Extender colocou na tabela base desta camada para assegurar que os dados espaciais da camada atendam os requisitos do sistema de referência espacial da camada.
- Eliminando os disparadores que são usados para atualizar a coluna espacial sempre que os dados do endereço forem incluídos, alterados ou removidos.

Quando os dados de endereço em uma linha da tabela são geocodificados, os dados espaciais resultantes serão colocados na mesma linha. Portanto, se a linha for excluída, os dados do endereço e os dados espaciais serão excluídos ao mesmo tempo. Os disparos não excluem os dados espaciais. Quando o procedimento armazenado é processado, as informações sobre a camada são removidas da exibição do catálogo DB2GSE.GEOMETRY_COLUMNS.

Autorização:

O ID de usuário com o qual este procedimento armazenado é chamado deve conter um dos seguintes privilégios ou autoridades:

- Para uma camada da tabela:
  - Autoridade SYSADM ou DBADM no banco de dados que contém a tabela base desta camada.
  - O privilégio CONTROL ou ALTER nesta tabela.
- Para uma camada da exibição:

– O privilégio SELECT na tabela base que contém (1) os dados de endereço que serão geocodificados para esta camada e (2) os dados espaciais que resultarem da geocodificação.

Parâmetros:

Tabela 91. Parâmetros de Entrada para o Procedimento Armazenado db2gse.gse_unregist_layer.

Nome	Tipo de Dados	Descrição
layerSchema	VARCHAR(30)	Nome do esquema ao qual pertence a tabela especificada no parâmetro layerTable. Este parâmetro pode ser nulo. Se não for fornecido um valor para o parâmetro layerSchema, será assumido como padrão o ID do usuário com o qual o procedimento armazenado db2gse.gse_unregister_layer é chamado. Você deve especificar em letras maiúsculas qualquer nome de esquema, nome de tabela, nome de exibição, nome de coluna ou nome de camada atribuídos a um parâmetro.
layerTable	VARCHAR(128)	Nome da tabela que contém a coluna especificada no parâmetro layerColumn. Este parâmetro não pode ser nulo.
layerColumn	VARCHAR(128)	Nome da coluna espacial que foi definida como a camada da qual você deseja cancelar o registro. Este parâmetro não pode ser nulo. Somente uma camada pode ser especificada para o parâmetro layerColumn. Portanto, ao remover o registro de várias camadas em uma tabela ou exibição, será necessário executar este procedimento armazenado separadamente para cada camada.

Resultados:

Tabela 92. Parâmetros de Saída para o Procedimento Armazenado db2gse.gse_unregist_layer.

Nome	Tipo de Dados	Descrição
msgCode	INTEGER	Código associado às mensagens que o responsável pela chamada deste procedimento armazenado poderá retornar.
msgText	VARCHAR(1024)	Mensagem de erro concluída, conforme construída no servidor DB2 Spatial Extender.

Restrições::

Se uma coluna da exibição definida como camada da exibição estiver baseada numa coluna da tabela, que foi definida como uma camada da tabela, não será possível cancelar o registro desta camada da tabela até que o registro da camada da exibição seja cancelado.

Apêndice B. Exibições de Catálogos Reprovadas

Este tópico descreve as exibições de catálogos reprovadas.

Nota: Recomendação: Desenvolva todos os novos aplicativos com as exibições definidas no DB2 Spatial Extender Versão 8. Todos os aplicativos atuais também devem ser atualizados para utilizar as exibições definidas na Versão 8. Os aplicativos que se referem a tabelas de catálogos básicas não documentadas definidas na Versão 7 não mais funcionarão após a migração para a Versão 8 e devem ser modificados para utilizar as exibições de catálogos documentadas da Versão 8.

DB2GSE.COORD_REF_SYS

Ao ativar um banco de dados para operações espaciais, o DB2 Spatial Extender registra os sistemas de coordenadas que podem ser utilizados em uma tabela de catálogo. As colunas selecionadas nesta tabela compõem a exibição do catálogo DB2GSE.COORD_REF_SYS, que é descrita na tabela a seguir.

Tabela 93. Colunas na Exibição de Catálogo DB2GSE.COORD_REF_SYS

Nome	Tipo de Dados	Permite nulos?	Conteúdo
CSID	INTEGER	Sim	Identificador numérico inteiro para este sistema coordenado. Se o sistema de coordenadas foi criado utilizando a interface de administração V8, nenhum CSID será registrado e será utilizado nulo
CS_NAME	VARCHAR(64)	Não	Nome deste sistema coordenado.
AUTH_NAME	VARCHAR(256)	Sim	Nome da organização que compilou o sistema coordenado agrega à; por exemplo, o European Petroleum Survey Group (EPSG).
AUTH_SRID	INTEGER	Sim	Um identificador numérico designado para este sistema coordenado, pela organização especificada na coluna AUTH_NAME column.
DESC	VARCHAR(256)	Sim	Descrição deste sistema coordenado.
SRTEXT	VARCHAR(2048)	Não	Texto de anotação para este sistema coordenado.

DB2GSE.GEOMETRY_COLUMNS

Ao criar uma camada, o DB2 Spatial Extender registra-a gravando seu identificador e informações referentes a ela em uma tabela de catálogo. As colunas selecionadas nesta tabela compõem a exibição do catálogo DB2GSE.GEOMETRY_COLUMNS, que é descrita na tabela a seguir.

Tabela 94. Colunas na Exibição de Catálogo DB2GSE.GEOMETRY_COLUMNS

Nome	Tipo de Dados	Permite nulos?	Conteúdo
LAYER_CATALOG	VARCHAR(30)	Sim	NULL. Não há nenhum conceito de LAYER_CATALOG no DB2 Spatial Extender.

Tabela 94. Colunas na Exibição de Catálogo DB2GSE.GEOMETRY_COLUMNS (continuação)

Nome	Tipo de Dados	Permite nulos?	Conteúdo
LAYER_SCHEMA	VARCHAR(30)	Não	Esquema da tabela ou da exibição que contém a coluna que foi registrada como esta camada.
LAYER_TABLE	VARCHAR(128)	Não	Nome da tabela ou da exibição que contém a coluna que foi registrada como esta camada.
LAYER_COLUMN	VARCHAR(128)	Não	Nome da coluna que foi registrada como esta camada.
GEOMETRY_TYPE	INTEGER	Sim	Tipo de dados da coluna que foi registrada como esta camada. Se a coluna tiver um subtipo definido pelo usuário de qualquer um dos tipos de geometrias definidos pelo spatial extender, este valor será nulo
SRID	INTEGER	Não	Identificador do sistema de referência espacial usado para os valores na coluna que foi registrada como esta camada.
STORAGE_TYPE	INTEGER	Sim	NULL.

DB2GSE.SPATIAL_GEOCODER

Os geocoders disponíveis estão registrados em uma tabela do catálogo. As colunas selecionadas nesta tabela compõem a exibição do catálogo DB2GSE.SPATIAL_GEOCODER, que é descrita na tabela a seguir.

Tabela 95. Colunas na Exibição de Catálogo DB2GSE.SPATIAL_GEOCODER

Nome	Tipo de Dados	Permite Nulos?	Conteúdo
GCID	INTEGER	Não	Identificador numérico do geocoder.
GC_NAME	VARCHAR(64)	Não	Identificador de nome do geocoder.
VENDOR_NAME	VARCHAR(128)	Não	Nome do fornecedor do geocoder.
PRIMARY_UDF	VARCHAR(256)	Não	Nome completo do geocoder.
PRECISION_LEVEL	INTEGER	Não	O grau de coincidência dos dados fonte com os dados de referência correspondente a fim de ser processado com sucesso pelo geocoder.
VENDOR_SPECIFIC	VARCHAR(256)	Sim	Caminho e nome de um arquivo que um fornecedor pode utilizar para definir quaisquer parâmetros especiais suportados pelo geocoder.
GEO_AREA	VARCHAR(256)	Sim	Área geográfica contendo as localizações para serem geocodificadas.
DESCRIPTION	VARCHAR(256)	Sim	Descrição do geocoder.

DB2GSE.SPATIAL_REF_SYS

Ao criar um sistema de referência espacial, o DB2 Spatial Extender registra-o gravando seu identificador e informações referentes a ele em uma tabela de catálogo. As colunas selecionadas nesta tabela compõem a exibição do catálogo DB2GSE.SPATIAL_REF_SYS, que é descrita na tabela a seguir.

Tabela 96. Colunas na Exibição do Catálogo DB2GSE.SPATIAL_REF_SYS

Nome	Tipo de Dados	Permite Nulos?	Conteúdo
SRID	INTEGER	Não	Identificador definido pelo usuário para este sistema de referência espacial.
SR_NAME	VARCHAR(64)	Não	Nome deste sistema de referência espacial.
CSID	INTEGER	Não	Identificador numérico para o sistema coordenado que suporta este sistema de referência espacial.
CS_NAME	VARCHAR(64)	Não	Nome do sistema coordenado que suporta este sistema de referência espacial.
AUTH_NAME	VARCHAR(256)	Sim	Nome da organização que define os padrões para este sistema de referência espacial.
AUTH_SRID	INTEGER	Sim	O identificador que a organização especificada na coluna AUTH_NAME assinala para este sistema de referência espacial.
SRTEXT	VARCHAR(2048)	Não	Texto de anotação para este sistema de referência espacial.
FALSEX	FLOAT	Não	Um número que, quando subtraído de um valor negativo da coordenada X, deixa um número não-negativo (ou seja, um número positivo ou zero).
FALSEY	FLOAT	Não	Um número que, quando subtraído de um valor negativo da coordenada Y, permite um número não-negativo (ou seja, um número positivo ou zero).
XYUNITS	FLOAT	Não	Um número que quando multiplicado por uma coordenada X decimal ou por uma coordenada Y decimal produz um inteiro que pode ser armazenado como um item de dados de 32–bits.
FALSEZ	FLOAT	Não	Um número que, quando subtraído de um valor negativo da coordenada Z, permite um número não-negativo (ou seja, um número positivo ou zero).
ZUNITS	FLOAT	Não	Um número que quando multiplicado por uma coordenada Z decimal produz um inteiro que pode ser armazenado como um item de dados de 32–bits.
FALSEM	FLOAT	Não	Um número que, quando subtraído de uma medida negativa, permite um número não-negativo (ou seja, um número positivo ou zero).
MUNITS	FLOAT	Não	Um número que quando multiplicado por uma medida decimal produz um inteiro que pode ser armazenado como um item de dados de 32–bits.

Exibições de Catálogos Reprovadas

Apêndice C. Funções Espaciais Reprovadas

Este tópico descreve as funções que foram reprovadas. A tabela abaixo lista todas as funções espaciais reprovadas e as novas funções de substituição da Versão 8.

Tabela 97. As funções reprovadas e seus novos equivalentes.

Função reprovada	**Nova Função**
AsShape	ST_AsShape
GeometryFromShape	ST_Geometry
Is3D	ST_Is3D
IsMeasured	ST_IsMeasured
LineFromShape	ST_Linestring
LocateAlong	ST_FindMeasure
LocateBetween	ST_MeasureBetween
M	ST_M
MLine FromShape	ST_MultiLineString
MPointFromShape	ST_MultiPoint
MPolyFromShape	ST_MultiPolygon
PointFromShape	ST_Point
PolyFromShape	ST_Polygon
ShapeToSQL	ST_Geometry
ST_GeomFromText	ST_Geometry
ST_GeomFromWKB	ST_Geometry
ST_LineFromText	ST_LineString
ST_LineFromWKB	ST_LineString
ST_MLineFromText	ST_MultiLineString
ST_MLineFromWKB	ST_MultiLineString
ST_MPointFromText	ST_MultiPoint
ST_MPointFromWKB	ST_MultiPoint
ST_MPolyFromText	ST_MultiPolygon
ST_MPolyFromWKB	ST_MultiPolygon
ST_OrderingEquals	
ST_Point(Double, Double, db2gse.coordref)	ST_Point(Double, Double, Integer)
ST_PointFromText	ST_Point
ST_PolyFromText	ST_Polygon
ST_PolyFromWKB	ST_Polygon
ST_Transform(Double, Double, db2gse.coordref)	ST_Transform(ST_Geometry, Integer)
ST_SymmetricDiff	ST_SymDifference
Z	ST_Z

AsShape

Propósito:

O AsShape recebe um objeto geométrico e retorna um BLOB.

Formato:

db2gse.AsShape(g db2gse.ST_Geometry)

Resultados:

BLOB(1m)

GeometryFromShape

Propósito:

GeometryFromShape recebe um formato e um identificador de sistema de referência espacial para retornar um objeto de figura geométrica.

Formato:

db2gse.GeometryFromShape(ShapeGeometry Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_Geometry

Is3d

Propósito:

Is3d utiliza um objeto de geometria e retorna 1 se o objeto tiver coordenadas 3D; caso contrário, retorna 0.

Formato:

db2gse.Is3d(g db2gse.ST_Geometry)

Resultados:

Integer

IsMeasured

Propósito:

IsMeasured utiliza um objeto de geometria e retorna 1 se o objeto tiver medidas; caso contrário, retorna 0.

Formato:

db2gse.IsMeasured(g db2gse.ST_Geometry)

Resultados:

Integer

LineFromShape

Propósito:

LineFromShape recebe um formato do tipo ponto e um identificador do sistema de referência espacial e retorna um segmento de reta.

Formato:

db2gse.Line FromShape(ShapeLineString Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_LineString

LocateAlong

Propósito:

LocateAlong toma um objeto de figura geométrica e uma medida para retornar como um multiponto do conjunto de pontos encontrados na medida.

Se LocateAlong receber um ponto múltiplo e uma medida como entradas e se o ponto múltiplo não incluir essa medida, o LocateAlong retornará POINT EMPTY.

Formato:

db2gse.LocateAlong(g db2gse.ST_Geometry, measure Double)

Resultados:

db2gse.ST_Geometry

LocateBetween

Propósito:

O LocateBetween pega um objeto de figura geométrica e duas localizações de medida e retorna uma figura geométrica que representa o conjunto de caminhos desconectados entre as duas localizações de medida.

Formato:

db2gse.LocateBetween(g db2gse.ST_Geometry, measure Double, measure Double)

Resultados:

db2gse.ST_Geometry

M

Propósito:

M toma um ponto e retorna sua medida.

Formato:

db2gse.M(p db2gse.ST_Point)

Resultados:

Double

MLine FromShape

Propósito:

MLine FromShape recebe um formato do tipo segmento de reta e um identificador do sistema de referências espaciais e retorna um segmento de reta múltiplo.

Formato:

db2gse.MLineFromShape(ShapeMultiLineString Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiLineString

MPointFromShape

Propósito:

MPointFromShape recebe um formato do tipo ponto múltiplo e um identificador do sistema de referências espaciais e retorna um ponto múltiplo.

Formato:

db2gse.MPointFromShape(ShapeMultiPoint BLOB(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiPoint

MPolyFromShape

Propósito:

MPolyFromShape recebe um formato do tipo polígono múltiplo e um identificador do sistema de referências espaciais para retornar um polígono múltiplo.

Formato:

db2gse.MPolyFromShape(ShapeMultiPolygon Blob(1m), SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiPolygon

PointFromShape

Propósito:

PointFromShape recebe um formato do tipo ponto e um identificador do sistema de referências espaciais para retornar um ponto.

Formato:

db2gse.PointFromShape(db2gse.ShapePoint blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_Point

PolyFromShape

Propósito:

PolyFromShape recebe um formato do tipo polígono e um identificador do sistema de referências espaciais para retornar um polígono.

Formato:

db2gse.PolyFromShape (ShapePolygon Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_Polygon

ShapeToSQL

Propósito:

ShapeToSQL constrói um valor de db2gse.ST_Geometry, fornecendo sua representação de formato. O valor 0 de SRID é usado automaticamente.

Formato:

db2gse.ShapeToSQL(ShapeGeometry blob(1M))

Resultados:

db2gse.ST_Geometry

ST_GeomFromText

Propósito:

ST_GeomFromText recebe uma representação de um texto convencional e um identificador do sistema de referências espaciais e retorna um objeto de figura geométrica.

Formato:

db2gse.ST_GeomFromText(geometryTaggedText Varchar(4000), SRID db2gse.coordref)

Resultados:

db2gse.ST_Geometry

ST_GeomFromWKB

Propósito:

ST_GeomFromWKB recebe uma representação de binário convencional e um identificador do sistema de referências espaciais e retorna um objeto de figura geométrica.

Formato:

db2gse.ST_GeomFromWKB(WKBGeometry Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_Geometry

ST_LineFromText

Propósito:

ST_LineFromText recebe uma representação de texto convencional do tipo segmento de reta e um identificador do sistema de referências espaciais e retorna um segmento de reta.

Formato:

db2gse.ST_LineFromText(lineStringTaggedText Varchar(4000), SRID db2gse.coordref)

Resultados:

db2gse.ST_LineString

ST_LineFromWKB

Propósito:

ST_LineFromWKB recebe uma representação binária convencional do tipo segmento de reta e um identificador do sistema de referências espaciais e retorna um segmento de reta.

Formato:

db2gse.ST_LineFromWKB(WKBLineString Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_LineString

ST_MLineFromText

Propósito:

ST_MLineFromText recebe uma representação de texto convencional do tipo segmento de reta múltiplo e um identificador do sistema de referência espacial e retorna um segmento de reta múltiplo.

Formato:

db2gse.ST_MLineFromText(multiLineStringTaggedText String, SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiLineString

ST_MLineFromWKB

Propósito:

ST_MLineFromWKB recebe uma representação binária convencional do tipo segmento de reta múltiplo e um identificador do sistema de referências espaciais e retorna um segmento de reta múltiplo.

Formato:

db2gse.ST_MLineFromWKB(WKBMultiLineString Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiLineString

ST_MPointFromText

Propósito:

ST_MPointFromText recebe uma representação de texto convencional do tipo ponto múltiplo e um identificador do sistema de referências espaciais e retorna um ponto múltiplo.

Formato:

db2gse.ST_MPointFromText(multiPointTaggedText Varchar(4000), SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiPoint

ST_MPointFromWKB

Propósito:

ST_MPointFromWKB recebe uma representação binária convencional do tipo ponto múltiplo e um identificador do sistema de referências espaciais para retornar um ponto múltiplo.

Formato:

db2gse.ST_MPointFromWKB(WKBMultiPoint Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiPoint

ST_MPolyFromText

Propósito:

ST_MPolyFromText recebe uma representação de texto convencional do tipo polígono múltiplo e um identificador do sistema de referências espaciais e retorna um polígono múltiplo.

Essa função não pode receber como entrada um polígono múltiplo que contenha polígonos múltiplos com as mesmas coordenadas.

Formato:

db2gse.ST_MPolyFromText(multiPolygonTaggedText Varchar(4000), SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiPolygon

ST_MPolyFromWKB

Propósito:

ST_MPolyFromWKB recebe uma representação binária convencional do tipo polígono múltiplo e um identificador do sistema de referência espacial e retorna um polígono múltiplo.

Formato:

db2gse.ST_MPolyFromWKB(WKBMultiPolygon Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_MultiPolygon

ST_OrderingEquals

Propósito:

ST_OrderingEquals compara as duas figuras geométricas e retorna 1 (VERDADEIRO) se as figuras geométricas forem iguais e as coordenadas estiverem na mesma ordem; do contrário, retornará 0 (FALSO).

Formato:

db2gse.ST_OrderingEquals(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Resultados:

Integer

ST_Point

Propósito:

ST_Point retorna um ST_Point, dada uma coordenada x, coordenada y e referência espacial.

Formato:

db2gse.ST_Point(X Double, Y Double, SRID db2gse.coordref)

Resultados:

db2gse.ST_Point

ST_PointFromText

Propósito:

ST_PointFromText recebe uma representação de texto convencional do tipo ponto e um identificador do sistema de referências espaciais e retorna um ponto.

Formato:

db2gse.ST_PointFromText(pointTaggedText Varchar(4000), SRID db2gse.coordref)

Resultados:

db2gse.ST_Point

ST_PolyFromText

Propósito:

ST_PolyFromText recebe uma representação de texto convencional do tipo polígono e um identificador do sistema de referências espaciais e retorna um polígono.

Formato:

db2gse.ST_PolyFromText(polygonTaggedText Varchar(4000), SRID db2gse.coordref)

Resultados:

db2gse.ST_Polygon

ST_PolyFromWKB

Propósito:

ST_PolyFromWKB recebe uma representação binária convencional do tipo polígono e um identificador do sistema de referências espaciais para retornar um polígono.

Formato:

db2gse.ST_PolyFromWKB(WKBPolygon Blob(1M), SRID db2gse.coordref)

Resultados:

db2gse.ST_Polygon

ST_Transform

Propósito:

ST_Transform associa uma figura geométrica a um sistema de referências espaciais diferente do sistema de referências espaciais ao qual a figura geométrica está associada atualmente.

Formato:

db2gse.ST_Transform(g db2gse.ST_Geometry, SRID db2gse.coordref)

Resultados:

db2gse.ST_Geometry

ST_SymmetricDiff

Propósito:

O ST_SymmetricDiff toma dois objetos de figura geométrica e retorna um objeto de figura geométrica que é a diferença simétrica dos objetos de origem.

A função ST_SymmetricDiff retorna a diferença simétrica (a lógica booleana XOR de espaço) de duas figuras geométricas de intersecção que possuem a mesma dimensão. Se essas figuras geométricas forem iguais, o ST_SymmetricDiff retornará uma figura geométrica vazia. Se forem diferentes, uma parte de uma delas ou de

ambas ultrapassará a área de interseção. ST_SymmetricDiff retorna a parte ou as partes que não fazem interseção como uma coleção; por exemplo, como um polígono múltiplo.

Se ST_SymmetricDiff fornecer figuras geométricas de diferentes dimensões como entrada, ele retornará um nulo.

Formato:

db2gse.ST_SymmetricDiff(g1 db2gse.ST_Geometry, g2 db2gse.ST_Geometry)

Resultados:

db2gse.ST_Geometry

Z

Propósito:

Z toma um ponto e retorna sua coordenada Z.

Formato:

db2gse.Z(p db2gse.ST_Point)

Resultados:

Double

Referência Relacionada:
- "ST_AsShape" na página 354
- "ST_MeasureBetween, ST_LocateBetween" na página 435
- "ST_EnvIntersects" na página 383
- "ST_FindMeasure ou ST_LocateAlong" na página 389
- "ST_Geometry" na página 397
- "ST_Is3d" na página 410
- "ST_LineString" na página 422
- "ST_M" na página 425
- "ST_MultiLineString" na página 453
- "ST_MultiPoint" na página 455
- "ST_MultiPolygon" na página 456
- "ST_Point" na página 468
- "ST_Polygon" na página 478
- "ST_SymDifference" na página 487
- "ST_Transform" na página 498
- "ST_Z" na página 509

Avisos

É possível que a IBM não ofereça os produtos, serviços ou recursos discutidos neste documento em outros países. Consulte um representante IBM local para obter informações sobre produtos e serviços disponíveis atualmente em sua área. Qualquer referência a produtos, programas ou serviços IBM não significa que apenas produtos, programas ou serviços IBM possam ser utilizados. Qualquer produto, programa ou serviço funcionalmente equivalente, que não infrinja nenhum direito de propriedade intelectual da IBM, poderá ser utilizado em substituição a este produto, programa ou serviço. Entretanto a avaliação e verificação da operação de qualquer produto, programa ou serviço não-IBM são de responsabilidade do Cliente.

A IBM pode ter patentes ou solicitações de patentes pendentes relativas a assuntos tratados nesta publicação. O fornecimento desta publicação não garante ao Cliente nenhum direito sobre tais patentes. Pedidos de licença devem ser enviados, por escrito, para:

Gerência de Relações Comerciais e Industriais da IBM Brasil
Av. Pasteur 138-146
Botafogo
Rio de Janeiro - RJ
CEP 22290-240

Para pedidos de licença relacionados a informações de DBCS (Conjunto de Caracteres de Byte Duplo), entre em contato com o Departamento de Propriedade Intelectual da IBM em seu país ou envie pedidos de licença, por escrito, para:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

O parágrafo a seguir não se aplica a nenhum país em que tais disposições não estejam de acordo com a legislação local: A INTERNATIONAL BUSINESS MACHINES CORPORATION FORNECE ESTA PUBLICAÇÃO "NO ESTADO EM QUE SE ENCONTRA" SEM GARANTIA DE NENHUM TIPO, SEJA EXPRESSA OU IMPLÍCITA, INCLUINDO, MAS NÃO SE LIMITANDO ÀS GARANTIAS IMPLÍCITAS DE NÃO-VIOLAÇÃO, MERCADO OU ADEQUAÇÃO A UM DETERMINADO PROPÓSITO. Alguns países não permitem a exclusão de garantias expressas ou implícitas em certas transações; portanto, esta disposição pode não se aplicar ao Cliente.

Esta publicação pode incluir imprecisões técnicas ou erros tipográficos. Periodicamente, são feitas alterações nas informações aqui contidas; tais alterações serão incorporadas em futuras edições desta publicação. A IBM pode, a qualquer momento, aperfeiçoar e/ou alterar os produtos e/ou programas descritos nesta publicação, sem aviso prévio.

Referências nestas informações a Web sites não-IBM são fornecidas apenas por conveniência e não representam de forma alguma um endosso a estes Web sites. Os materiais contidos nestes Web sites não fazem parte dos materiais deste produto IBM e a utilização destes Web sites é de inteira responsabilidade do Cliente.

A IBM pode utilizar ou distribuir as informações fornecidas da forma que julgar apropriada sem incorrer em qualquer obrigação para com o Cliente.

Licenciados deste programa que desejam obter informações sobre este assunto com objetivo de permitir: (i) a troca de informações entre programas criados independentemente e outros programas (incluindo este), e (ii) a utilização mútua das informações trocadas, devem entrar em contato com:

Gerência de Relações Comerciais e Industriais da IBM Brasil
Av. Pasteur, 138-146
Botafogo
Rio de Janeiro, RJ
CEP: 22290-240

Tais informações podem estar disponíveis, sujeitas a termos e condições apropriadas, incluindo em alguns casos o pagamento de uma taxa.

O programa licenciado descrito neste documento e todo o material licenciado disponível são fornecidos pela IBM sob os termos do Contrato com o Cliente IBM, do Contrato de Licença do Programa Internacional IBM ou de qualquer outro contrato equivalente.

Todos os dados de desempenho aqui contidos foram determinados em um ambiente controlado. Portanto, os resultados obtidos em outros ambientes operacionais podem variar significativamente. Algumas medidas podem ter sido tomadas em sistemas de nível de desenvolvimento e não há garantia de que tais medidas serão iguais em sistemas geralmente disponíveis. Além disso, algumas medidas podem ter sido estimadas por extrapolação. Os resultados reais podem variar. Os usuários deste documento devem verificar os dados aplicáveis para o seu ambiente específico.

As informações relativas a produtos não-IBM foram obtidas junto aos fornecedores dos produtos, de seus anúncios publicados ou de outras fontes disponíveis publicamente. A IBM não testou estes produtos e não pode confirmar a precisão do desempenho, compatibilidade nem qualquer outra reivindicação relacionada a produtos não-IBM. Dúvidas sobre a capacidade de produtos não-IBM devem ser encaminhadas diretamente a seus fornecedores.

Todas as declarações relacionadas aos objetivos e intenções futuras da IBM estão sujeitas a alterações ou cancelamento sem aviso prévio e representam apenas metas e objetivos.

Estas informações podem conter exemplos de dados e relatórios utilizados nas operações diárias de negócios. Para ilustrá-lo da forma mais completa possível, os exemplos podem incluir nomes de indivíduos, empresas, marcas e produtos. Todos os nomes são fictícios e qualquer semelhança com nomes e endereços utilizados por uma empresa real é mera coincidência.

LICENÇA DE COPYRIGHT:

Estas informações podem conter programas aplicativos de exemplo na linguagem fonte, que ilustram as técnicas de programação em diversas plataformas operacionais. Você pode copiar, modificar e distribuir estes programas de exemplo sem a necessidade de pagar à IBM, com objetivos de desenvolvimento, utilização, marketing ou distribuição de programas aplicativos em conformidade com a interface de programação do aplicativo para a plataforma operacional para a qual os programas de exemplo são criados. Estes exemplos não foram testados

completamente em todas as condições. Portanto, a IBM não pode garantir ou confirmar a confiabilidade, manutenção ou função destes programas.

Cada cópia ou parte deste exemplo de programa ou qualquer trabalho derivado deve incluir um aviso de copyright com os dizeres:

© (*nome da sua empresa*) (*ano*). Partes deste código são derivadas dos Programas de Exemplo da IBM Corp. © Copyright IBM Corp. *_digite o ano ou anos_*. Todos os direitos reservados.

Marcas Comerciais

Os termos a seguir são marcas comerciais da International Business Machines Corporation nos Estados Unidos e/ou em outros países e foram utilizados em pelo menos um dos documentos da biblioteca de documentação do DB2 UDB.

ACF/VTAM
AISPO
AIX
AIXwindows
AnyNet
APPN
AS/400
BookManager
C Set++
C/370
CICS
Database 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Information Integrator
DB2 Query Patroller
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eServer
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
iSeries
LAN Distance
MVS
MVS/ESA
MVS/XA
Net.Data
NetView
OS/390
OS/400
PowerPC
pSeries
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/400
SQL/DS
System/370
System/390
SystemView
Tivoli
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WebSphere
WIN-OS/2
z/OS
zSeries

Os termos a seguir são marcas comerciais ou marcas de serviço de outras empresas e foram utilizados em pelo menos um dos documentos da biblioteca de documentação do DB2 UDB:

Microsoft, Windows, Windows NT e o logotipo Windows são marcas comerciais da Microsoft Corporation nos Estados Unidos e/ou em outros países.

Intel e Pentium são marcas comerciais da Intel Corporation nos Estados Unidos e/ou em outros países.

Java e todas as marcas comerciais baseadas em Java são marcas da Sun Microsystems, Inc. nos Estados Unidos e/ou em outros países.

UNIX é uma marca registrada do The Open Group nos Estados Unidos e em outros países.

Outros nomes de empresas, produtos ou serviços podem ser marcas comerciais ou marcas de serviço de terceiros.

Índice Remissivo

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

Z

Entrando em Contato com a IBM

Nos Estados Unidos, ligue para qualquer um dos seguintes números para entrar em contato com a IBM:

- 1-800-IBM-SERV (1-800-426-7378) para atendimento ao cliente
- 1-888-426-4343 para conhecer as opções de serviço disponíveis
- 1-800-IBM-4YOU (426-4968) para Departamento de Marketing e Vendas do DB2

No Canadá, ligue para qualquer um dos seguintes números para entrar em contato com a IBM:

- 1-800-IBM-SERV (1-800-426-7378) para atendimento ao cliente
- 1-800-465-9600 para conhecer as opções de serviços disponíveis
- 1-800-IBM-4YOU (1-800-426-4968) para o departamento de marketing e vendas do DB2

No Brasil, ligue para o seguinte número para entrar em contato com a IBM:

- 0-800-7014-262 para informações gerais

Para localizar um escritório da IBM em seu país ou região, acesse o Directory of Worldwide Contacts da IBM na Web no endereço http://www.ibm.com/planetwide

Informações sobre o Produto

As informações relacionadas aos produtos DB2 Universal Database estão disponíveis por telefone ou através da World Wide Web no endereço http://www.ibm.com/software/data/db2/udb

Este site contém as informações mais recentes sobre a biblioteca técnica, pedido de manuais, downloads de produtos, newsgroups, FixPacks, notícias e links para recursos da Web.

Se você mora no Brasil, ligue para o Centro de Atendimento a Clientes:

- 0-800-7014-262 para solicitar produtos ou obter informações gerais.
- 0-800-7014-850 - FAX para solicitar publicações.

Para obter informações sobre como entrar em contato com a IBM fora dos Estados Unidos, acesse a página Worldwide da IBM no endereço www.ibm.com/planetwide

Impresso em Brazil

S517-7368-01

Spine information:

IBM® DB2® Spatial Extender e Geodetic Extender

Referência e Guia do Usuário do DB2 Spatial Extender e Geodetic Extender

Versão 8.2