PostGIS

Antonio Espín Herranz

Contenidos

- Introducción
- Tipos de datos en PostGIS
- Funciones de PostGIS

Introducción

- Bases de datos espaciales:
  - Son aquellas desarrolladas con el uso de datos basados en coordenadas en mente.
  - Lo novedoso es que estos datos no son tratados de la forma tradicional.
  - Existen tipos, funciones y mecanismos de optimización internos creados expresamente para facilitar su procesamiento.
  - De esta forma podemos trabajar sin problemas con puntos, líneas o áreas en un sistema de coordenadas.

- **GIS** o SIG → Geographical Information System / Sistemas de información geográfica.
  - Tratamiento de datos espaciales.

Introducción

- **Postgis**: Extensión de PostGre para habilitar los datos espaciales en las bases de datos de PostGre.

- La extensión Postgis la tenemos que activar en una BD para poder trabajar con datos geográficos.
  - ***CREATE EXTENSION postgis;***
    - *La extensión hay que instalarla en la BD que vayamos a utilizar*

Características de Postgis

- **Almacenamiento de datos espaciales** : almacene diferentes tipos de datos espaciales, como puntos, líneas, polígonos y multigeometrías, tanto en datos 2D como 3D .
- **Indexación espacial** : buscar y recuperar rápidamente datos espaciales en función de su ubicación.
- **Funciones espaciales** : una amplia gama de funciones espaciales que le permiten filtrar y analizar datos espaciales, medir distancias y áreas , intersecar geometrías, almacenar en búfer y más.
- **Procesamiento de geometría** : herramientas para procesar y manipular datos geométricos, como simplificación , conversión y generalización.
- **Soporte de datos ráster** : almacenamiento y procesamiento de datos ráster , como datos de elevación y datos meteorológicos.
- **Geocodificación y geocodificación inversa** : Funciones para geocodificación y geocodificación inversa.
- **Integración** : PostGIS se integra con otras herramientas de terceros como QGIS , GeoServer , MapServer , ArcGIS, Tableau.

Introducción

- Enlace a postgis:
  - https://postgis.net/

- **MANUAL ONLINE**: https://postgis.net/docs/

- Versión en OCT 2024: **3.4.2 / 3.5**

- Podemos consultar la versión con las funciones:
  - select postgis_version();
  - SELECT PostGIS_Full_Version();

- Versiones alternativas de postgis instaladas:
  - SELECT * FROM pg_available_extensions WHERE name = 'postgis';

- Esta extensión:
  - Añade soporte de objetos geográficos a PostGre.
  - Permite utilizar objetos GIS incluyendo soporte para índices Gist y funciones básicas para el análisis de objetos GIS.
  - Cumple con las especificaciones OpenGIS y podemos trabajar con los siguientes tipos de datos: puntos, líneas, polígonos, multilíneas, multipuntos y colecciones geométicas.

Introducción

- Con las BD espaciales surgieron otros estándares:
  - SFSQL → Simple Features for SQL
  - Formatos principales:
    - WKT → Well Know Text
    - WKB → Well Know Binary
  - PostGIS implementa SFSQL y lo amplía con SQL/MM: soporte para datos hasta 4 dimensiones (3DM, 3DZ y 4D)

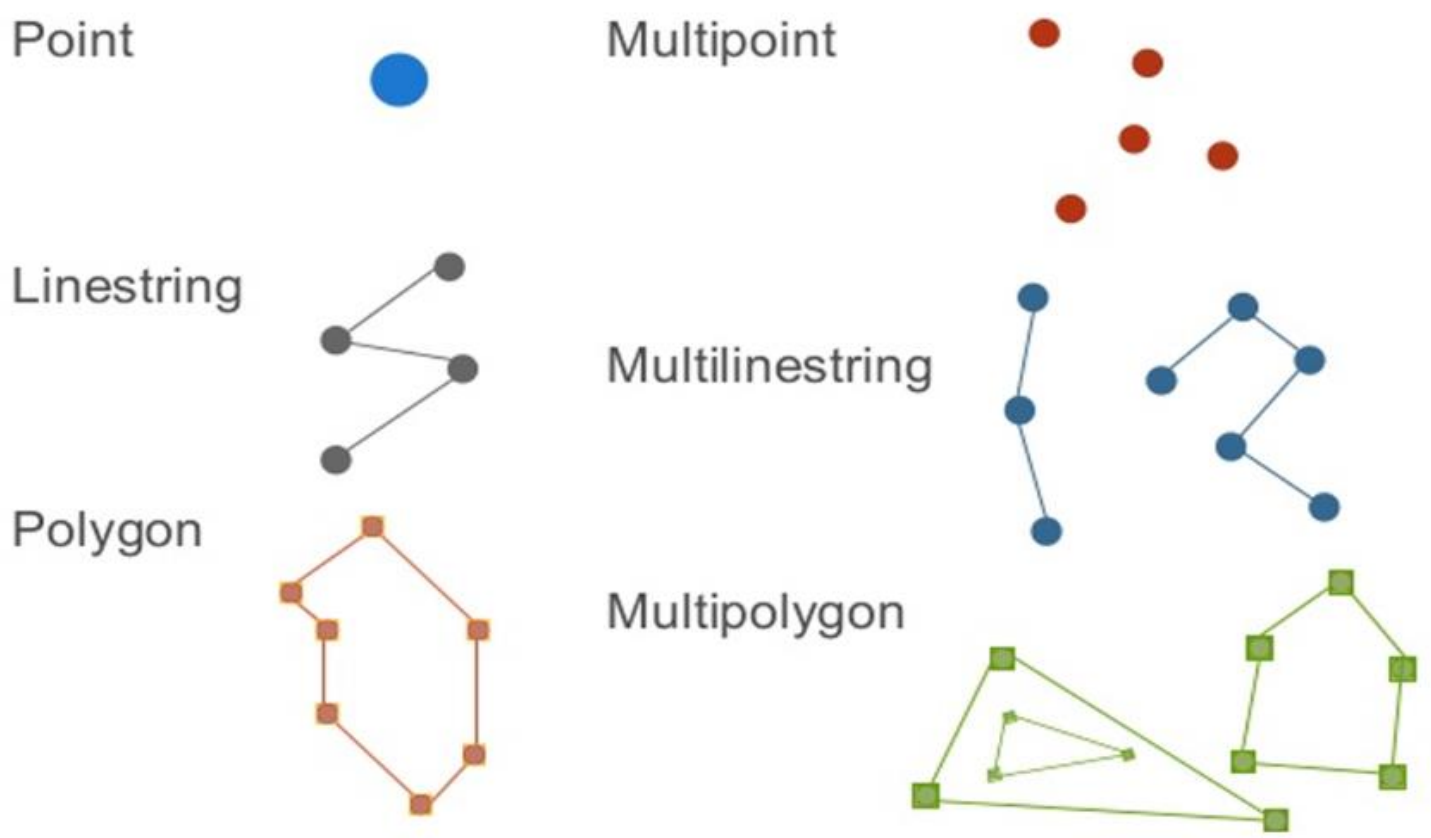
Introducción

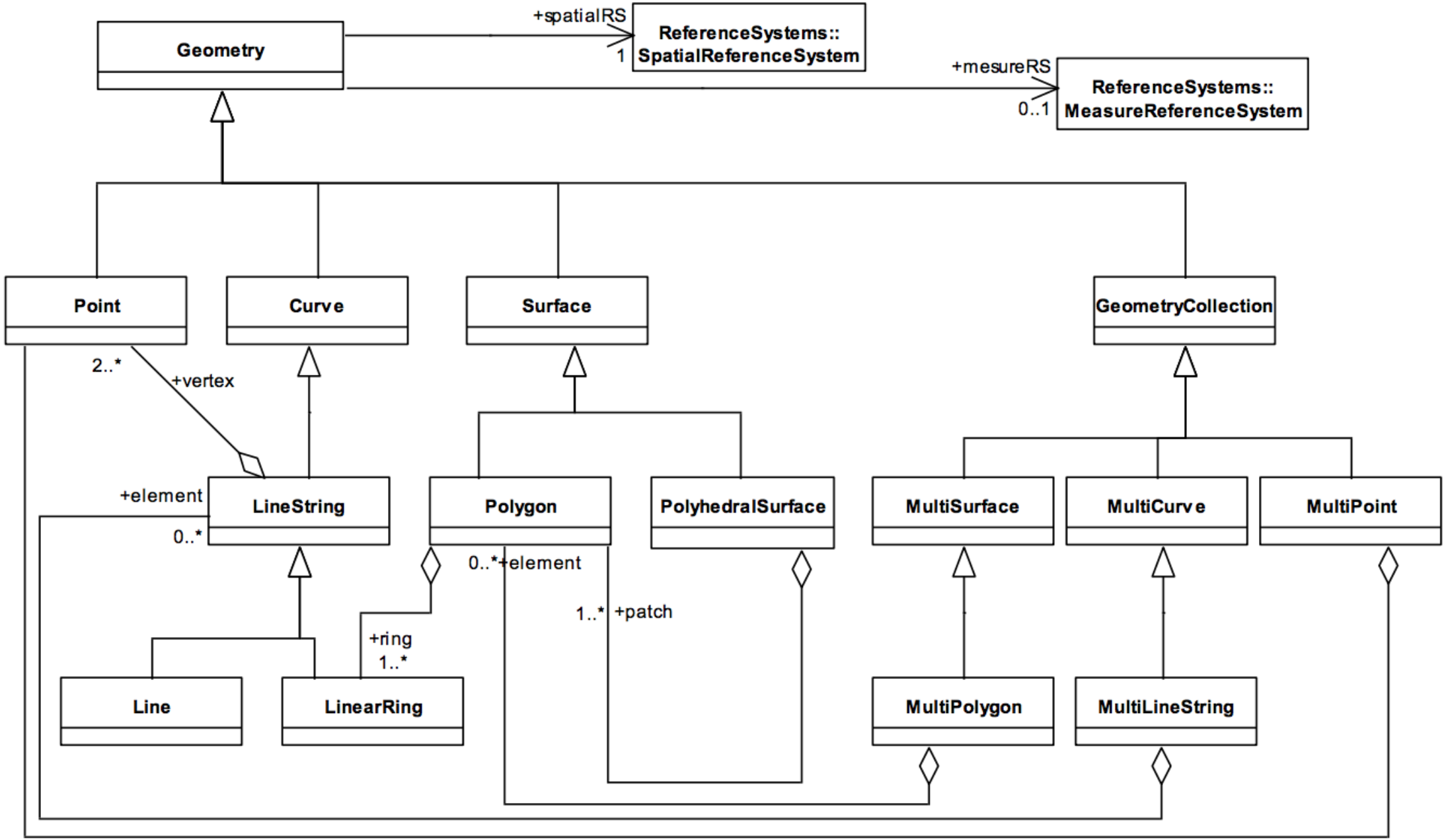
- Las geometrías se almacenan como un reglón en una tabla en una columna de tipo geometría.
  - Por convención esta columna se suele llamar **geom**.

- PostGIS soporta distintos tipos de datos relacionados con geometrías.

Tipos de datos en PostGIS

- Point
- Multipoint
- Linestring
- Multilinestring
- Polygon
- Multipolygon




Geometry
+spatialRS
1
ReferenceSystems::
SpatialReferenceSystem
+mesureRS
0..1
ReferenceSystems::
MeasureReferenceSystem
Point
Curve
Surface
GeometryCollection
2..*
+vertex
+element
0..*
LineString
Polygon
PolyhedralSurface
MultiSurface
MultiCurve
MultiPoint
0..*+element
1..* +patch
+ring
1..*
Line
LinearRing
MultiPolygon
MultiLineString

Funciones

- Tipos de funciones:
  - Cálculo de atributos geométricos:
    - ST_Area
    - ST_Perimeter
    - ST_Length
  - Construcción de geometrías:
    - St_MakePoint
    - ST_Makeline
    - ST_MakePolygon
    - ST_GeomFromText
    - AddGeometryColumn / DropGeometryColumn

Funciones 2

- Tipos de funciones:
    - Funciones para el análisis espacial:
        - ST_Buffer
        - ST_SymDifference
        - ST_Intersection
        - ST_Union
        - ST_Centroid
        - ST_Envelope

Funciones 3

- Tipos de funciones:
    - Funciones booleanas para relaciones entre geometrías:
        - ST_Contains
        - ST_Intersects
        - ST_Overlaps
        - ST_Touches
        - ST_Covers

Funciones 4

- Tipos de funciones:
  - Funciones relativas a sistemas de coordenadas y posición:
    - ST_Translate
    - ST_Transform
    - ST_RID
    - ST_SetSRID
  - Verificar geometrías:
    - ST_GeometryType
    - ST_IsValid

Funciones para cálculo de atributos geométricos

- **ST_Area**
  - Devuelve el área de un polígono o multipolígono. Hay que pasar la columna que contiene la geometría. Si la geometría no tiene formato de polígono devuelve 0.
  - Select st_area(geom) as área from datos.polígonos;

- **ST_Perimeter**
  - Como la anterior, pero devolviendo el perímetro. Si la geometría no tiene formato de polígono devuelve 0.
  - Select st_perimeter(geom) as área from datos.polígonos;

- **ST_Length**
  - Devuelve la longitud bidimensional de una geometría de tipo línea o multilínea. Si la geometría no tiene formato de polígono devuelve 0.
  - Select st_length(geom) as longitud from datos.polígonos;

Funciones para la construcción de geometrías

- **St_makepoint**(x, y):
  - Crea una geometría de tipo punto
  - X → longitud, y → latitud
  - Crea un punto de 2, 3, 4 dimensiones
  - SELECT ST_MakePoint(-71.1043443253471, 42.3150676015829);

- **St_makeline**(geom)
  - Crea una geometría de tipo línea.
  - Se puede crear a partir de un array de puntos.
  - SELECT ST_MakeLine(ARRAY[ST_MakePoint(5,2), ST_MakePoint(4,5), ST_MakePoint(8,10)]);

Funciones para la construcción de geometrías

- **St_MakePolygon**
  - Permite construir una geometría de tipo polígono en base a una geometría de tipo línea. **La geometría lineal debe ser cerrada**, es decir, que su nodo inicial y final sean coincidentes.

    - SELECT ST_MakePolygon(ST_GeomFromText('LINESTRING(10 10, 15 10, 20 15, 10 15, 10 10)'));

- **St_GeomFromText**
  - Devuelve un objeto geométrico en base a una expresión en formato WKT (Well Known Text) que lo defina.
    - SELECT ST_GeomFromText('POINT(20 20)')
    - SELECT ST_GeomFromText('LINESTRING(-5 10, 10 10, 10 15, 20 40, -10 20)')
    - SELECT ST_GeomFromText('POLYGON((10 10, 15 10, 10 15, 5 5, 10 10))')

Funciones para la construcción de geometrías

- **AddGeometryColumn**
- **DropGeometryColumn**
  - Permiten crear o eliminar la columna que contiene el objeto geográfico de una tabla. Debe especificarse el esquema de la tabla, el nombre del campo, el SRID o sistema de coordenadas, el tipo de geometría y sus dimensiones.
  - Parámetros:
    - Esquema, tabla, nombre del campo, SRID o sistema de coordenadas,
    - SELECT AddGeometryColumn('datos', 'nuevospuntos', 'geom', 25830, 'POINT', 2);

    - Esquema, tabla, nombre de campo
    - SELECT DropGeometryColumn('datos', 'nuevospuntos', 'geom');

Funciones para el análisis espacial

- **ST_Buffer**
  - Devuelve una geometría que representa todos aquellos puntos comprendidos dentro de un radio determinado desde un objeto geométrico dado. Los cálculos se basarán en el SRID de la geometría.
  - SELECT ST_Buffer(geom, 750) FROM datos.lineas;

- **ST_SymDifference**.
  - Devuelve una geometría que representa las partes de los objetos de la tabla A y la tabla B que no se intersecan. Recibe el nombre de diferencia simétrica porque da exactamente lo mismo pasar como primer argumento una u otra tabla, es decir, ST_SymDifference(A,B) = ST_SymDifference(B,A).
  - SELECT ST_SymDifference(parques.geom, ST_Buffer(papeleras.geom, 500))
  - FROM datos.poligonos as parques, datos.puntos as papeleras;

Funciones para el análisis espacial

- **ST_Intersection**
  - Devuelve la geometría que representa los puntos comunes entre dos geometrías de entrada. Es importante, en este caso, recordar que al tratarse de puntos en común entre ambas capas, se evalúan A y B de la misma manera, es decir, ST_Intersection(A, B) = ST_Intersection(B, A)

  - SELECT ST_Intersection(parques.geom,
  - ST_Buffer(papeleras.geom, 500))
  - FROM datos.poligonos as parques,
  - datos.puntos as papeleras;

Funciones para el análisis espacial

- **ST_Union**
  - Devuelve una geometría que representa la unión de dos tablas con datos espaciales.
  - SELECT ST_Union(lineas.geom, poligonos.geom) FROM datos.lineas, datos.poligonos;

- **ST_Centroid**
  - Devuelve el centroide calculado a partir de las geometrías de entrada.
  - SELECT ST_Centroid(poligonos.geom) FROM datos.poligonos;

- **ST_Envelope**
  - Devuelve una geometría que representa el bounding box de las geometrías que contiene la tabla.
  - SELECT ST_Envelope(lineas.geom) FROM datos.lineas;

Funciones booleanas para relaciones entre geometrías

- **ST_Contains**
  - Retorna un valor booleano True para todas aquellas geometrías de B contenidas en A que si, y solo si, que ninguno de sus puntos se encuentran fuera de los límites de A. En caso contrario retorna el valor booleano False.
  - SELECT ST_Contains(poligonos.geom, lineas.geom) FROM datos.lineas, datos.poligonos;

- **ST_Intersects**
  - Retorna el valor booleano True si las geometrías de A y B comparten algún punto en común. Es decir, si se sobreponen en algún punto del espacio.
  - SELECT ST_Intersects(poligonos.geom, lineas.geom) FROM datos.lineas, datos.poligonos;

- **ST_Overlaps**
  - Retorna el valor booleano True si la geometría B se sobrepone a la geometría A, pero B sobrepasa los límites de A. Es decir, si ocupan parte del mismo espacio en las mismas dimensiones, pero sin llegar a contenerse la una a la otra.
  - SELECT ST_Overlaps(poligonos_a.geom, poligonos_b.geom) FROM datos.poligonos_a, datos.poligonos_b;

Funciones booleanas para relaciones entre geometrías

- **ST_Touches**
  - Devuelve True si las geometrías de A y B tienen algún punto de sus bordes / límites en común. Para puntos, aplica si sus coordenadas coinciden con algún punto situado en el límite de un polígono.
  - SELECT ST_Touches(poligonos.geom, lineas.geom) FROM datos.lineas, datos.poligonos;

- **ST_Covers**
  - Retorna True si ningun punto de B se encuentra fuera de A, es decir, si el área que ocupa A es capaz de cubrir todos los elementos geométricos por completo de B.
  - SELECT ST_Covers(poligonos.geom, lineas.geom) FROM datos.lineas, datos.poligonos;

Funciones relativas a sistemas de coordenadas y posición

- **ST_Translate**
  - Permite desplazar o mover la geometría, tanto en 2 dimensiones como en 3 dimensiones. Debemos pasar como argumentos la geometría y los valores de desplazamiento en cada eje.
  - SELECT ST_SetSRID(ST_Point(-3.7018, 40.3185), 4326);

- **ST_Transform**
  - Permite reproyectar las geometrías a un sistema de referencia de coordenadas definido. En nuestro caso reproyectamos del SRID original (EPSG: 25830) de los polígonos de la tabla al EPSG:4326:
  - SELECT ST_Translate(geom, -0.05, 0.25) FROM datos.poligonos;

Funciones relativas a sistemas de coordenadas y posición

- **ST_SRID**
  - Permite consultar cuál es el SRID de una geometría o conjunto de geometrías. Devuelve una columna en la que muestra el valor del código EPSG del SRID:
  - SELECT ST_Transform(geom, 4326) FROM datos.poligonos;

- **ST_SetSRID**
  - Permite establecer / asignar el SRID o Sistema de Referencia de Coordenadas de una geometría o conjunto de geometrías. En este caso estamos asignando el EPSG:4326 (WGS 84) al punto con sus coordenadas definidas:
  - SELECT ST_SRID(geom) FROM datos.poligonos;

Funciones para verificar geometrías

- **ST_GeometryType**
  - Permite conocer el tipo de geometría de un objeto.
  - Retorna un valor que define la tipología dentro de PostGIS, como: ST_Point, ST_Linestring, ST_Polygon, ST_Multipoint...
    - SELECT ST_GeometryType(geom) as tipo FROM datos.poligonos;

- **ST_IsValid**
  - Permite conocer si una geometría es válida. Retorna un valor booleano por cada geometría.
    - SELECT ST_IsValid(geom) as validez FROM datos.poligonos;

Otras funciones: F. de salida

- Es común querer ver los datos de nuestras tablas en una forma que podamos entender o exportarlos para tratarlos en otros lados.
- Estas funciones nos permiten, a partir de un tipo de dato geométrico, **obtener su representación** en una variedad bastante amplia de **formatos**: **binario, texto, GeoJSON, GML, SVG…**

- SELECT ST_AsText('POINT(0 0)'); -- Muestra nuestro punto en WTK (Well-Known Text).

- SELECT ST_AsEWKT('0101000000000000000000000000000000000000'); -- Lo mismo que antes, pero en este caso hemos pasado cómo se representa el punto si lo obtenemos de la tabla directamente.

- SELECT ST_AsGeoJSON('LINESTRING(1 2 3, 4 5 6)'); -- Muestra esta multilínea de un espacio de tres dimensiones en formato GeoJSON.

- SELECT ST_AsSVG('POLYGON((0 0,0 1,1 1,1 0,0 0))'); -- Muestra el polígono como un path data de SVG, que es el formato en el que se especifica cómo se dibuja una imagen SVG. Para más información podéis consultar https://www.w3.org/TR/SVG/paths.html#PathDataBNF

- SELECT ST_AsBinary('POLYGON((0 0,0 1,1 1,1 0,0 0))'::geometry); -- Devuelve la representación en binario del polígono (Well-Known Binary).

- **::geometry** → es un casting para convertir a tipo de dato.

Otras funciones: F. de construcción

- Como su nombre indica, permiten **crear objetos geométricos** a partir de una entrada, que puede estar representada en una amplia variedad de formatos. Si las funciones de salida eran la serialización entonces las de construcción serían las de **deserialización**.

- SELECT ST_AsEWKT(ST_GeomFromText('LINESTRING(0 0, 1 1, 1 2, 3 2)'));
- -- Crea una multilínea que pasa por los puntos (0, 0), (1, 1), (1, 2) y (3, 2).

- SELECT ST_AsText(ST_GeomFromGeoJSON('{"type":"LineString","coordinates":[[0,0],[1,1],[1,2],[3,2]]}'));
- -- Crea la misma línea, pero expresando la entrada en formato GeoJSON.

- SELECT ST_AsEWKT(ST_LineFromMultiPoint('MULTIPOINT(0 0, 1 1, 1 2, 3 2)'));
- -- Misma línea, pero a partir de un conjunto de puntos.

Otras funciones: F. de Acceso

- Nos permiten obtener **datos de un objeto concreto**, como el tipo de geometría, el número de puntos que tiene, la longitud, el área… Muchas de estas funciones no sirven para todos los tipos de datos, por ejemplo: tiene sentido saber si una multilínea es cerrada (si su origen y destino es el mismo) pero no aplica a un único punto que siempre se considera como cerrado.

- SELECT GeometryType('POINT(1 2 3)'::geometry);
- -- Devuelve el tipo concreto de geometría que es. En este caso un punto.

- SELECT ST_Dimension('GEOMETRYCOLLECTION(LINESTRING(1 1,0 0),POINT(0 0))’);
- -- Devuelve la dimensión de la geometría. Los puntos tienen dimensión 0, las líneas 1, los polígonos 2 y las colecciones tienen la mayor dimensión entre todos sus componentes.

- SELECT ST_IsClosed('LINESTRING(0 0, 1 1)’);
- -- Devuelve si la multilínea es cerrada.

- SELECT ST_AsText(ST_ExteriorRing('POLYGON((1 1, 2 5, 7 5, 7 1, 1 1), (3 3, 3 4, 5 4, 4 3, 3 3))'));
- -- Devuelve el anillo exterior del polígono.

Otras funciones: F. de Edición

- En vez de crear nuevos datos desde cero podemos querer basarnos en alguno ya existente y modificarlo para que se ajuste a nuestras necesidades. Los casos más típicos son los de **rotar o escalar una figura**.

- SELECT ST_AsText(ST_Rotate('LINESTRING (50 160, 50 50, 100 50)', pi()/2));
- -- Rota la multilínea que le hemos pasado en 90 grados en sentido antihorario.

- SELECT ST_AsText(ST_Scale('LINESTRING(1 2 3, 1 1 1)', 1, 2, 3));
- -- Escala la multilínea para que mantenga sus coordenadas en el eje X, se duplique en el Y y se triplique en el eje Z.

- SELECT ST_AsText(ST_AddPoint('LINESTRING(0 0 1, 1 1 1)', 'POINT (1 2 3)'));
- -- Añade un nuevo punto a una multilínea.

- SELECT ST_AsText(ST_Force2D('LINESTRING(0 0 1, 1 1 1)'));
- -- Fuerza la geometría a tener solo dos coordenadas, lo que permite adaptarse al estándar de OGC.

Otras funciones: F. de medidas y relaciones

- Algunas de las funciones más importantes que tenemos nos permiten **relacionar los datos** con su entorno, tanto con medidas individuales como con otros elementos.

- SELECT ST_Distance('POINT (1 1)', 'LINESTRING(0 0, 2 0)’);
- -- Calcula la distancia en dos dimensiones entre dos elementos.

- SELECT ST_3DDistance('POINT (1 1 1)', 'LINESTRING(0 0 0, 2 0 0)’);
- -- Calcula la distancia, pero esta vez en un espacio de 3 dimensiones.

- SELECT ST_Within('POINT(2 2)'::geometry, 'LINESTRING(1 1, 3 3)'::geometry);
- -- Determina si todos los puntos del primer elemento se encuentran dentro del segundo.

- SELECT ST_Equals('LINESTRING(0 0, 10 10)', 'LINESTRING(0 0, 5 5, 10 10)’);
- -- Determinar si dos elementos son iguales, es decir, si se cumple que el primero está dentro del segundo y el segundo dentro del primero. Esto quiere decir que, como en el ejemplo, no es necesario que estén creados igual.

- SELECT ST_Length('LINESTRING(0 0, 1 0, 1 1)’);
- -- Devuelve la longitud total de una multilínea.

Otras funciones: F. procesamiento geométrico

- Permiten **crear figuras** a partir de propiedades de uno o más elementos.

- SELECT ST_AsText(ST_Buffer('POINT(100 90)', 50));
- -- Devuelve el área que representa todos los puntos que están a una distancia de 50 unidades del punto.

- SELECT ST_AsText(ST_Union('POLYGON((1 2, 2 4, 8 5, 9 4, 9 2, 1 2))'::geometry, 'POLYGON((8 4, 8 6, 10 6, 10 4, 8 4))'::geometry));
- -- Devuelve el polígono fruto de la unión de los dos parámetros.

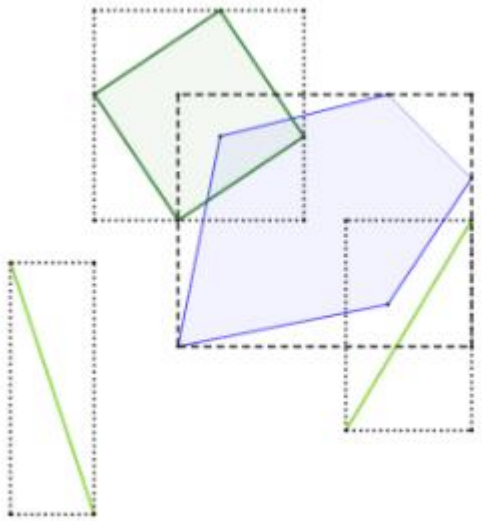
Indices

- Índices enfocados especialmente a tratar con estos datos.
- **R-Tree** es el utilizado por PostGIS (y algunos otros sistemas como Oracle Spatial).

- Los **árboles R** (o *R trees*) son la estructura de datos en la que se apoya *PostGIS* para crear y mantener sus índices.

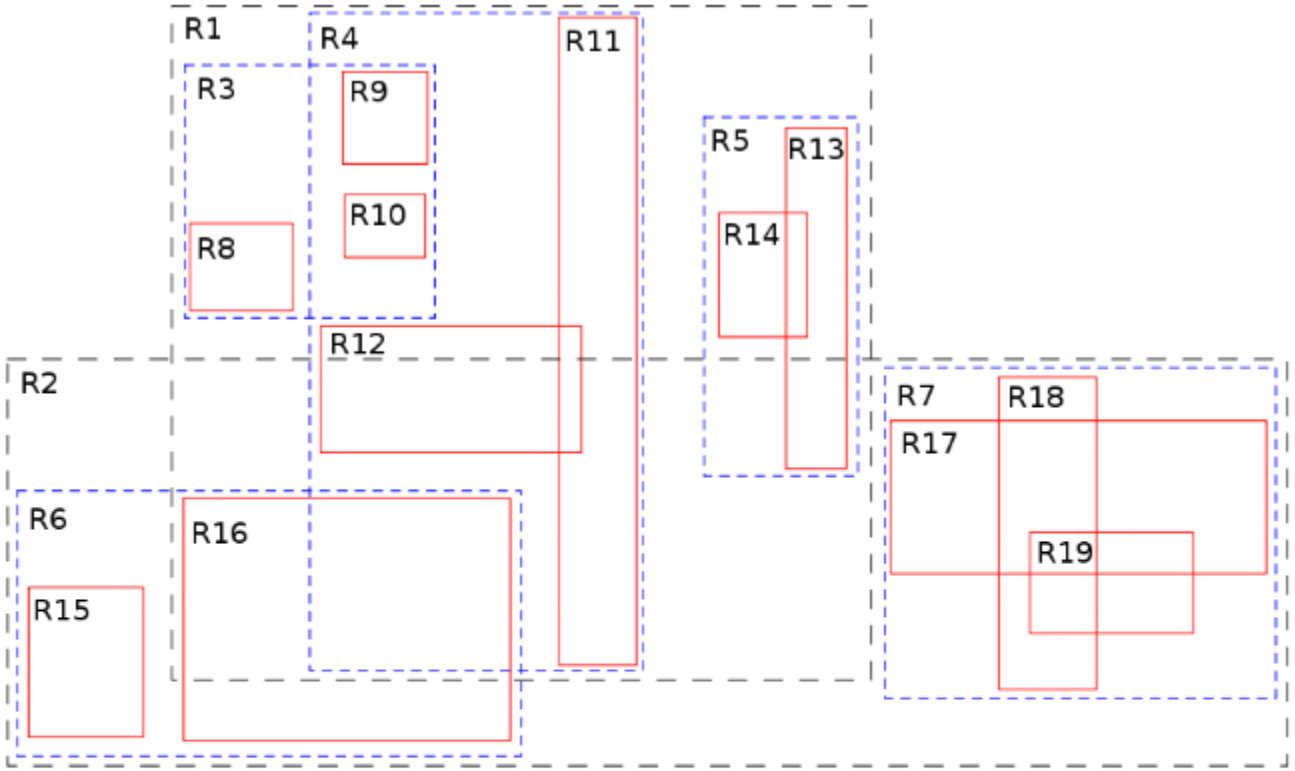
- Se basa en dos conceptos: usar **rectángulos delimitadores de superficie mínima** (*Minimum Bounding Rectangle* o *MBR* en inglés) y **agrupar** los objetos cercanos en áreas cada vez más pequeñas.

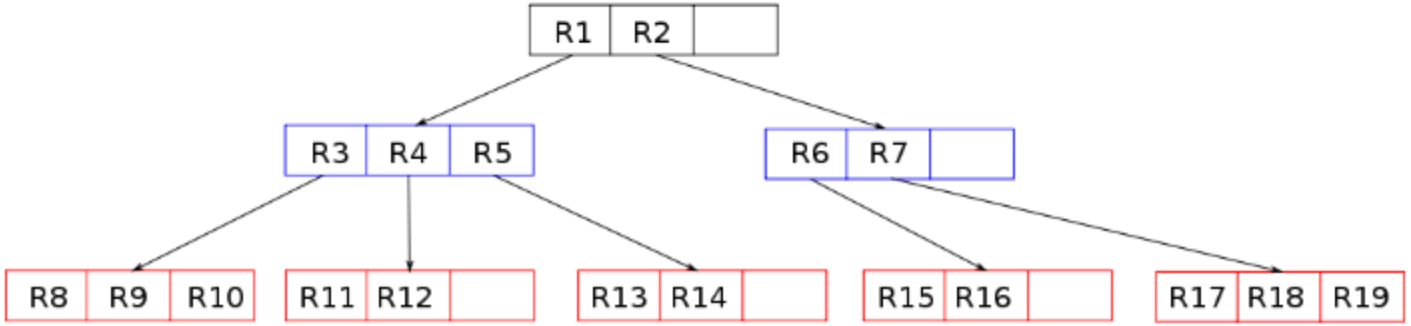
Índices



- Encierra cada objeto en su: MBR para comprobar si están contenidos o se tocan.

Índices





- Un **árbol R** (la R viene de rectángulo) se trata de un **árbol balanceado de búsqueda** que divide el espacio en rectángulos cada vez más pequeños que agrupan todos los objetos dentro de ellos y que pueden llegar a superponerse.

Enlaces

- Adictos al trabajo:
  - https://adictosaltrabajo.com/2019/09/17/postgis-para-entender-las-bases-de-datos-espaciales/
- Coordenadas:
  - https://www.linuxhispano.net/2014/11/03/almacenamiento-y-tratamiento-de-datos-geolocalizados-en-postgresql-con-postgis/
- Video:
  - https://youtu.be/Uu1DjCVRmLQ
- Funciones PostGIS
  - https://www.geomapik.com/analisis-gis/postgis-analisis-espacial-funciones/
- Hydrology GIS Data
  - https://data.world/city-of-bloomington/9eafa188-3bfe-41b8-a518-d942df0503ca

Enlaces

- SRID: https://www-alibabacloud-com.translate.goog/blog/an-overview-of-srid-and-coordinate-system_597004?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=rq#:~:text=An%20SRID%20indicates%20coordinates%20of,where%20srtext%20~*%20'china’%3B