

Bases de Datos Espaciales

Nicolás Del Piano

Bases de Datos Avanzadas

- ▶ Pensando Espacialmente
- ▶ Bases de Datos Espaciales
 1. Definición
 2. DBMS
- ▶ Hacia un Modelo de Datos Espacial
 1. Objetos en el espacio
 2. Espacio
 3. Organizando el espacio
 4. Tipos de Datos y Álgebras Espaciales
 5. Consultas
- ▶ Indexación Espacial
 1. R-Tree
 2. Estructuras
- ▶ Estado del Arte

¿Dónde está esto? ¿Cuánto tardo para llegar?

Hoy en día tenemos grandes herramientas para responder este tipo de preguntas.

Google Maps, Virtual Earth, MapQuest, Yahoo, etc...

Yendo más allá, las organizaciones han descubierto que estas herramientas son un gran recurso para analizar patrones de datos.

Por ejemplo, una popular empresa de pizzas, trazando los recorridos de las direcciones de los "pizza lovers", puede encontrar fácilmente dónde inaugurar su nuevo local.

Pensar Espacialmente, ¿en un DBMS relacional?

La información es almacenada y controlada en los DBMS relacionales.

Si bien son muy populares y responden a la mayoría de las consultas, la visión real es que muchas de ellas son incapaces de manejar datos espaciales, o son poco amigables.

¿Por qué?

El rol tradicional de un DBMS es y ha sido almacenar información relacionada al mundo de los negocios.

Pensar Espacialmente, ¿en un DBMS relacional?

Los datos que residen en esas grandes bases de datos es simple (nombres, saldos, direcciones, etcétera).

Por ejemplo, una consulta como "listar todos los empleados cuyo sueldo es mayor a X" puede ser simple y eficientemente respondida, aún en bases de datos gigantes.

Pero, ¿qué sucede si quiero saber: "listar todos los empleados que residan a menos de 10 km de la compañía?"

En los DBMS relacionales, es posible contestar estas preguntas... pero es totalmente **ineficiente**.

En conclusión, en muchas áreas existe la necesidad de administrar información *geométrica, geográfica o espacial*, es decir, información relacionada con el **espacio**.

Las **Bases de Datos Espaciales** pueden ayudarnos.

Sistemas de Información Geográfica.

Bases de Datos Multimedia.

Imágenes satelitales.

Modelo de circuitos integrados de gran tamaño.

Censos.

Ciencias Ambientales.

Astronomía.

¿Quién se beneficia?

Usuario de celular: ¿Dónde está la siguiente estación de servicio? ¿Hay una camino a casa?

Médico: Basado en el siguiente MRI, ¿hemos tratado a alguien con el mismo resultado?

Biólogo molecular: ¿Está la topología de la biosíntesis del gel aminoácido en el genoma descubierto en otra secuencia de la base de datos?

Astrónomo: Buscar todas las galaxias dentro de dos minutos de arcos de quásares.

Climatólogo: Testear y verificar mi nuevo modelo climático.

¿Qué se busca?

Tener soporte para representar este tipo de tablas:

| Id | País | Geografía |
|----|-----------|---|
| 1 | Argentina |  |
| 2 | Brasil |  |

Y además brindar herramientas que permitan comparar eficientemente datos como los de la tercer columna.

Bases de Datos Espaciales

Definición₁

Es una base de datos que es optimizada para almacenar y consultar información relacionada a objetos en el espacio, incluyendo puntos, líneas y polígonos.

Definición₂

Es una colección de datos espaciales y no espaciales que están interrelacionados.

Deben permitir la descripción de los objetos espaciales mediante tres características:

- ▶ **Atributos:** qué es un objeto de acuerdo a sus características.
- ▶ **Localización:** conocer dónde está el objeto y qué lugar ocupa.
- ▶ **Topología:** mejorar la interpretación semántica del contexto y establecer jerarquías.

Definición

- ▶ *Un SDBMS es un DBMS.*
- ▶ *Ofrece datos espaciales (SDTs) en su modelo de datos y su lenguaje de consulta.*
- ▶ *Soporta tipos de datos espaciales en su implementación, proveyendo al menos indexación espacial y algoritmos eficientes para joins espaciales.*

Un SDBMS es un DBMS.

Suena trivial; refleja el hecho de que la información espacial o geométrica está, en práctica, conectada con la información "no-espacial".

El SDBMS tiene propiedades adicionales para manejar datos espaciales.

Tipos de datos espaciales.

POINT, LINE, REGION, etcétera.

Proveen una abstracción fundamental para modelar la estructura de entidades geométricas en el espacio, así como sus relaciones, propiedades y operaciones.

Objetivos de un SDBMS

Integrar la representación y manipulación de datos geométricos y espaciales con los datos no espaciales en un nivel lógico.

Proveer un soporte eficiente para almacenar y procesar datos a nivel físico.

Requerimientos de un SDBMS

El lenguaje de consulta debe incorporar nuevas funciones sobre componentes geométricos.

Utilizar métodos de acceso eficientes.

Incorporar nuevos algoritmos para procesar consultas que satisfacen combinaciones de restricciones.

La representación lógica debe ser extendida a datos geométricos.

Capacidad de manejar grandes colecciones de objetos geométricos.

HACIA UN MODELO ESPACIAL

Modelando Datos Espaciales

Asumiendo aplicaciones 2-d y GIS, existen dos aspectos importantes que necesitan ser representados:

- ▶ **Objetos en el Espacio:** distinguir entre entidades espaciales de acuerdo a su representación geométrica.
- ▶ **Espacio:** describir cada punto del espacio, es decir, tener una idea de la representación del espacio mismo.

Modelando Datos Espaciales

Los datos espaciales tienen las siguientes desventajas:

- ▶ Estructura compleja.
- ▶ Las SBD tienden a ser muy grandes.
- ▶ No existe un álgebra estándar.
- ▶ Operadores no cerrados.
- ▶ Costo computacional elevado.

Es por ello que hay que implementar algoritmos y estructuras de datos que optimicen la manipulación de los mismos.

Objetos en el Espacio

Objetos únicos e individuales.

Para modelarlos, las abstracciones fundamentales son el *punto*, la *línea* y la *región* (polígono).



Punto

Un punto representa el aspecto geométrico de un objeto para el cual lo único relevante es su localización en el espacio.

Por ejemplo, una ciudad puede ser modelada como un punto en un mapa.



Una línea representa es la abstracción básica del movimiento a través del espacio, o conexiones en espacio.

Por ejemplo, ríos, carreteras, cables de teléfono, etcétera.

Usualmente representadas como *polilíneas*, i.e, secuencias de segmentos.

Región o polígono

Una región es una abstracción para algo que tenga una extensión en el espacio 2-d.

Por ejemplo un país, un lago, un parque nacional, etcétera.

Una región puede tener huecos y puede ser conformada por distintos objetos del espacio.



Hay que caracterizar los puntos del espacio.

De esta forma, podemos modelar mapas temáticos que indiquen por ejemplo, la división de un país en provincias.

Existen dos importantes instancias en las colecciones de objetos relacionadas espacialmente (espacio).

Son las *particiones* y las *redes*.

Particiones

Una partición puede ser vista como un conjunto de regiones que son disjuntas.

La adyacencia en este caso tiene un interés particular; pares de regiones que tienen frontera común.

Pueden ser usadas para representar mapas.



Una red puede ser vista como un grafo embebido en el plano.

Está compuesta por un conjunto de objetos puntos (nodos) y un conjunto de objetos líneas (aristas).

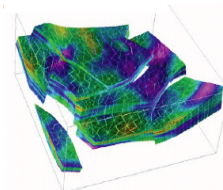
Pueden ser usadas para representar elementos geográficos como carreteras, ríos, transportes públicos, etcétera.



Otras abstracciones

Hemos mencionado hasta aquí las más comunes.

Obviamente, existen otras como las *particiones anidadas* (e.g. un país dividido en provincias, y a su vez cada provincia en estados), o los modelos de terrenos digitales, para representar relieves.



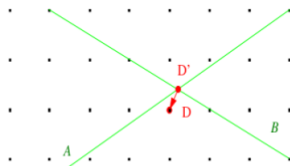
Organizando el Espacio

La geometría Euclidiana no es adecuada como una base para modelar en las bases de datos espaciales.

- ▶ El espacio Euclídeo es **continuo**, i.e., los puntos son representados como un par de números reales:
 $p = (x, y) \in \mathbf{R}^2$.
- ▶ Pero la computadora trabaja en forma **discreta**.
 - ▶ \Rightarrow el espacio es representado por un *raster discreto*.

Ejemplo: Intersección de dos líneas

- ▶ El punto de intersección es redondeado al punto de la grilla más cercano.
- ▶ Una prueba para determinar si el punto de intersección está en una de las líneas determina un resultado falso.



Organizando el Espacio

Solución: Definir una base geométrica discreta para modelar el espacio, así también como para lidiar con su implementación.

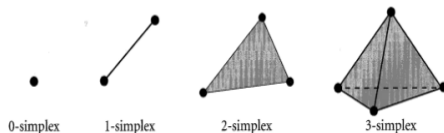
Dos intentos para una base geométrica:

- ▶ **Simplicial complexes** (Frank & Kuhn)
- ▶ **Realms** (Güting & Schneider)

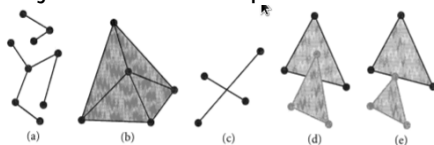
Organizando el Espacio: Simplicial Complexes

Conceptos básicos de **simplicial complexes**

- **d-simplex:** Un objeto de dimensión d



- **Simplicial Complex:** Un conjunto finito de simplices tales que la intersección de dos simplices cualquiera en el conjunto es un componente usado en los simplices.

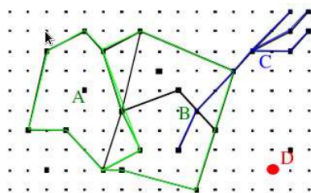


Organizando el Espacio: Realm

Realm: Un conjunto finito de puntos y segmentos de líneas definidos sobre una grilla tal que:

- ▶ cada punto o punto final de los segmentos es un punto en la grilla
- ▶ cada punto final de un segmento es un punto del realm
- ▶ ningún punto del realm se encuentra dentro de un segmento
- ▶ dos segmentos no se intersectan salvo en sus puntos finales

Realm provee una descripción completa de la geometría (puntos y líneas).



Tipos de Datos y Álgebras Espaciales

Las básicas abstracciones espaciales pueden ser embebidas en un DBMS existente usando tipos abstractos de datos.

- ▶ **Tipos de Datos Espaciales** encapsulan
 - ▶ la estructura de un objeto espacial, e.g., región
 - ▶ operaciones sobre esas estructuras
- ▶ **Álgebra Espacial**, es una colección de datos espaciales con operaciones relacionadas
 - ▶ Completitud y clausura deben ser propiedades del álgebra

Álgebra ROSE

ROSE (RObust Spatial Extension, Güting & Schneider)

Es un álgebra espacial con tipos de datos espaciales basados en el modelo Realm (esto es, objetos compuestos por elementos realm)

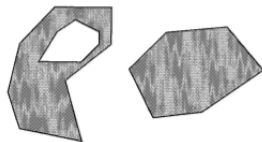
Tipos de datos de ROSE: puntos, líneas y regiones.



a points value



a line value



a region value

Álgebra ROSE: Operaciones

$\text{EXT} = \{\text{líneas, regiones}\}$ $\text{GEO} = \{\text{puntos, líneas, regiones}\}$

- ▶ Predicados espaciales para relaciones topológicas:
 - ▶ **inside**: $\text{geo} \times \text{regions} \rightarrow \text{bool}$
 - ▶ **intersect, meets**: $\text{ext1} \times \text{ext2} \rightarrow \text{bool}$
 - ▶ **adjacent, encloses**: $\text{regions} \times \text{regions} \rightarrow \text{bool}$
- ▶ Operaciones que devuelven tipos de datos espaciales atómicos:
 - ▶ **intersection**: $\text{lines} \times \text{lines} \rightarrow \text{points}$
 - ▶ **intersection**: $\text{regions} \times \text{regions} \rightarrow \text{regions}$
 - ▶ **plus, minus**: $\text{geo} \times \text{geo} \rightarrow \text{geo}$
 - ▶ **contour**: $\text{regions} \rightarrow \text{lines}$

Álgebra ROSE: Operaciones

$\text{EXT} = \{\text{líneas, regiones}\}$ $\text{GEO} = \{\text{puntos, líneas, regiones}\}$

- ▶ Operadores espaciales que devuelven números:
 - ▶ **dist**: $\text{geo1} \times \text{geo2} \rightarrow \text{real}$
 - ▶ **perimeter, area**: $\text{regions} \rightarrow \text{real}$
- ▶ Operadores espaciales en conjuntos de objetos:
 - ▶ **sum**: $\text{set(obj)} \times (\text{obj} \rightarrow \text{geo}) \rightarrow \text{geo}$
 - ▶ Alguna función espacial agregada como por ejemplo la suma de las áreas de las provincias determinan el área de un país.
 - ▶ **closest**: $\text{set(obj)} \times (\text{obj} \rightarrow \text{geo1}) \times \text{geo2} \rightarrow \text{set(obj)}$
 - ▶ Determina dentro de un conjunto de objetos aquellos cuyo atributo espacial tiene mínima distancia.
 - ▶ Otras operaciones complejas...

Álgebra ROSE: Propiedades

Estos ejemplos son suficientes para ver qué tipo de operaciones están disponibles en un álgebra espacial.

Algunas propiedades que debe tener el álgebra espacial son:

- ▶ Extensibilidad
- ▶ Completitud
- ▶ ¿Uno o más tipos?
- ▶ Operaciones de conjuntos

Entre las operaciones espaciales, podemos distinguir las *relaciones*.

- ▶ Topológicas: adjacent, inside, disjoint. Son invariantes dentro de transformaciones topológicas como scaling, rotation, translation.
- ▶ De dirección: above, below, north_of.
- ▶ Métricas: $\text{distance} < 100$.

Seis relaciones topológicas válidas entre dos regiones simples (sin huecos, conectadas): disjoint, in, touch, equal, cover, overlap.

Integrando la Geometría en el Modelo de Datos

Los tipos de datos espaciales pueden ser embebidos en cualquier modelo de datos, e.g., el modelo relacional.

El modelo de datos del DBMS debe ser extendido al nivel de tipos de datos atómicos, como strings y enteros.

La idea básica es representar **objetos espaciales** con objetos (del modelo de datos del DBMS) con al menos un atributo espacial.

Los objetos espaciales son tuplas con al menos un atributo espacial.

```
relation estados (ename: STRING; area:REGION; epop:  
INTEGER)
```

```
relation ciudades (cname: STRING; centro:POINT;  
ext:REGION; cpop: INTEGER)
```

```
relation rios (rname: STRING; ruta:LINE)
```

Dos problemas principales:

- ▶ Conectar las operaciones del álgebra espacial (incluyendo los predicados de relaciones espaciales) a las facilidades de un lenguaje de consulta de un DBMS.

Los operadores fundamentales son:

- ▶ Spatial selection
 - ▶ Spatial join
 - ▶ Overlay, fusion, ...
- ▶ Proveer una representación gráfica de la información espacial (o sea, los resultados de la consulta) y además una entrada gráfica para los valores en las consultas.

Consultas: Selección Espacial

Selección Espacial: retorna aquellos objetos que satisfacen un predicado espacial en la consulta.

Todas las ciudades de Argentina

```
SELECT cname FROM ciudades c  
WHERE c.centro inside Argentina.area
```

Todas las grandes ciudades en un radio de 100km de Rosario

```
SELECT cname FROM ciudades c  
WHERE dist(c.centro, Rosario.centro) < 100  
      AND c.pop > 500k
```

Consultas: Join Espacial

Join Espacial: un join que compara cualesquiera dos objetos que fueron *joinados* basado en un predicado que toma los valores espaciales de ambos.

Para cada río de Argentina, encontrar todas las ciudades dentro de 50 kms

```
SELECT r.rname, c.cname,  
length(intersection(r.ruta, c.area))  
FROM rios r, ciudades c  
WHERE r.ruta intersects Argentina.area  
      AND dist(r.ruta, c.area) < 50
```

Overlay: retorna las regiones que son el resultado de sobreposicionar dos particiones.

Fusion: es una forma especial de agrupación. Para cada grupo obtenido, se forma la unión de todos los valores de un atributo espacial.

Voronoi: computa de una colección de objetos puntos S la correspondiente colección de objetos región. Para cada punto p , la región consiste de los puntos del plano más cercanos a p , y que no estén más cercanos a otro punto de S .

Consultas: I/O Gráfica

¿Cómo determinar Argentina en los ejemplos anteriores(input), o cómo mostrar `intersection(ruta,Argentina.area)` o `rio.ruta(output)`?

Requerimientos para consultas espaciales:

- ▶ Tipos de datos espaciales.
- ▶ Muestreo gráfico de los resultados de consulta.
- ▶ Combinación gráfica de varios resultados de consultas.
- ▶ Herramientas para interactuar con gráficos de entrada y resultados.
- ▶ Facilidad para verificar el contenido de una muestra.

Existen dos estándares de representación:

- ▶ **WKT** (Well Known Text)
- ▶ **WKB** (Well Known Binary)

Formato de Datos Espaciales: WKT

Esta representación es una codificación en formato ASCII para describir objetos espaciales.

Es usada por PostgreSQL en PostGIS, y también en Google Maps.

Podemos describir puntos, multipuntos, líneas, polígonos, multipolígonos, puntos en 3 dimensiones, etcétera.

Sintaxis:

- ▶ Punto: `POINT(30 50)`
- ▶ Línea: `LINESTRING(1 1, 5 5, 10 10)`
- ▶ Polígono: `POLYGON ((0 0, 10 0, 10 10, 0 10, 0 0), (20 20, 20 40, 40 40, 40 20, 20 20))`

Formato de Datos Espaciales: WKB

WKB utiliza enteros sin signo de un byte, enteros sin signo de cuatro bytes, y números de ocho bytes de doble precisión (formato IEEE 754). Un byte son ocho bits.

Por ejemplo, un valor WKB que corresponde a un POINT(11) consiste en esta secuencia de 21 bytes (cada uno representado aquí por dos dígitos hexadecimales):

010100000000000000000000F03F000000000000F03F

Orden de byte 01 → LE o BE

Tipo WKB 01000000 → tipo de objeto

X 00000000000000F03F

Y 00000000000000F03F

Más rápida que WKT, pero menos entendible para el usuario.

INDEXACIÓN ESPACIAL

Indexación Espacial

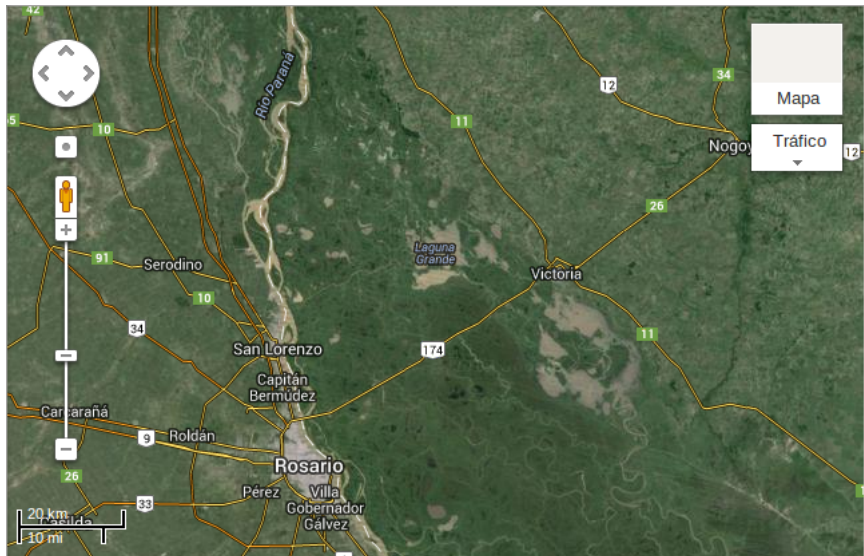
Organiza el espacio y los objetos en él de forma tal que sólo son considerados la parte del espacio y los objetos que sean relevantes a una consulta.

Es un mecanismo para decrementar el número de búsquedas.

El objetivo es tratar de no visitar y tocar cada vez a todos los n elementos en la BD.

Los índices espaciales nos permiten optimizar la recuperación.

Indexación Espacial



Indexación Espacial

El uso principal es para la selección espacial.

- ▶ Pero es usado también para otras operaciones como join espacial o encontrar el objeto más cercano...

Dos enfoques principales:

- ▶ Mapear objetos espaciales al espacio 1-D y utilizar técnicas estándar de indexación, como B-trees
- ▶ Estructuras de indexación espacial dedicadas, como R-trees

Indexación Espacial: R-Tree

R-Tree

Es una estructura de datos similar a los B-Trees, con la diferencia que se utilizan para métodos de acceso espacial. Es decir, podemos indexar coordenadas por ejemplo.

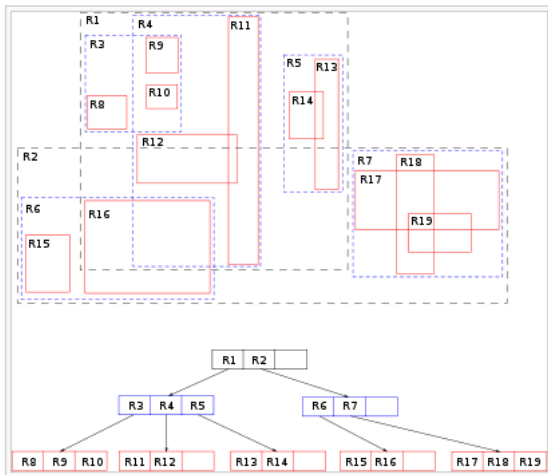
Cada nodo tiene un número variable de entradas, y cada entrada almacena dos datos: una forma de identificar a un nodo hijo y el conjunto límite de todas las entradas de ese nodo hijo.

Búsqueda: recursiva desde la raíz, se verifica si el rectángulo de la consulta se solapa con algún rectángulo del nodo.

Insertión: partiendo de la raíz, se usa una heurística para seleccionar el nodo candidato.

Indexación Espacial: R-Tree

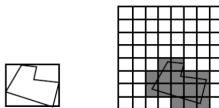
R-Tree



Indexación Espacial

La idea fundamental de la indexación espacial es la aproximación.

- ▶ Aproximación continua
- ▶ Aproximación por grilla



Para procesar una consulta, se procede a la estrategia de **filtrado y refinamiento**.

1. Filtrado: retorna un conjunto (donde está la solución) de los valores posibles que satisfacen un predicado
2. Refinado: Para cada candidato, se refina la búsqueda chequeando la geometría

Indexación Espacial

Las estructuras espaciales son diseñadas para almacenar puntos (para objetos puntos) o rectángulos (líneas y regiones). Las operaciones sobre esas estructuras son inserción, borrado y pertenencia.

Consultas típicas:

- ▶ Puntos
 - ▶ Consulta de rango: todos los puntos dentro de un rectángulo
 - ▶ Vecino más cercano: el punto más cercano a un punto consultado
 - ▶ Distancia: desde un punto, enumerar los puntos que están a cierta distancia
- ▶ Rectángulos
 - ▶ Intersección
 - ▶ Contenido

Indexación Espacial: Estructuras

- ▶ Una estructura de indexación espacial dedicada organiza los objetos en **cubetas**.
- ▶ Cada cubeta tiene asociada una **región**, que es, una parte del espacio que contiene todos los objetos almacenados en esa cubeta.
- ▶ Para estructuras de datos de puntos, las regiones son disjuntas
 - ▶ el espacio es particionado y cada punto pertenece a solo una partición (cubeta)

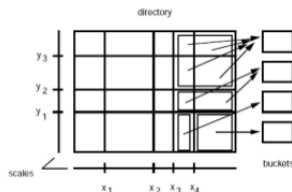


- ▶ Para estructuras con rectángulos, las cubetas pueden solaparse

Indexación Espacial: Estructura de Puntos

Estructuras espaciales para puntos: en k dimensiones son representadas como una tupla $t = (x_1, \dots, x_k)$

Índice de Grilla: Estructura de indexación espacial para puntos (Nievergelt, Hinterberger y Sevcik 84)

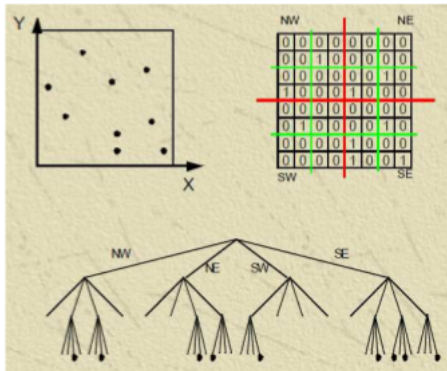


- ▶ El directorio es un arreglo k -dimensional cuyas entradas son punteros a las cubetas.
- ▶ Todos los puntos en las celdas son almacenados en cubetas apuntando a la entrada de directorio correspondiente.

Indexación Espacial: Estructura de Puntos

quad-tree:

Divide el espacio de datos en forma recursiva en 4 cuadrantes (NO, NE, SO, SE)

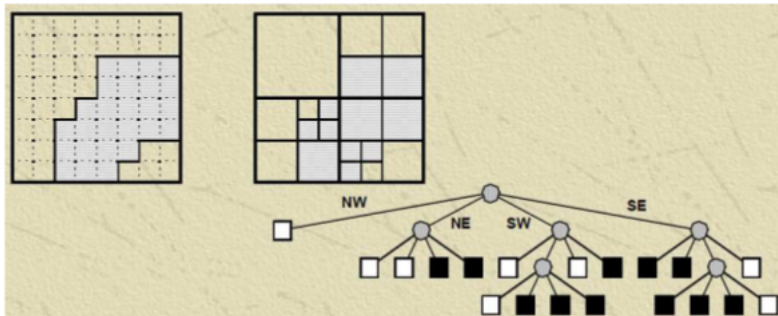


Indexación Espacial: Estructura de Puntos

quad-tree:

Hay diferentes algoritmos para procesar puntos, líneas, polígonos (i.e., distintos tipos de nodos, algoritmos de consultas).

Es usado muy frecuentemente en GIS comerciales para comprimir, almacenar y manipular imágenes raster.



Indexación Espacial: Estructura de Rectángulo

Dos enfoques principales:

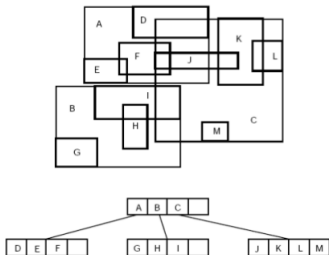
- ▶ Solapamiento de regiones
- ▶ Clipping (Recorte)

Indexación Espacial: Estructura de Rectángulo

Solapamiento de regiones

Se abandona el espacio particionado y las regiones pueden solaparse, e.g., **R-tree**.

- ▶ Ventaja: El objeto espacial (clave) está en una sola cubeta.
- ▶ Desventaja: Muchos caminos terminan en un solapamiento de cubetas.

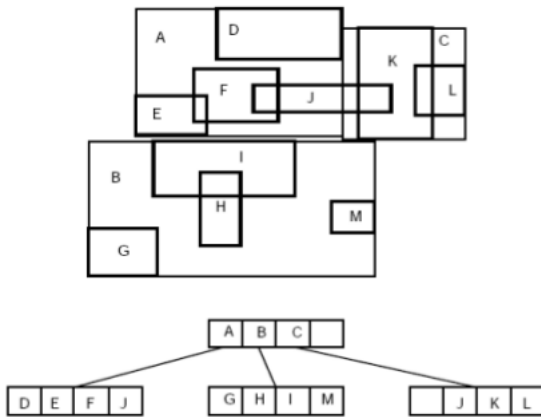


Indexación Espacial: Estructura de Rectángulo

Recorte (Clipping)

Las cubetas son disjuntas, pero los rectángulos son cortados en varias piezas, e.g., **R⁺-tree**.

- ▶ Ventaja: Menos ramas en el árbol
- ▶ Desventaja: Múltiples entradas para un objeto espacial

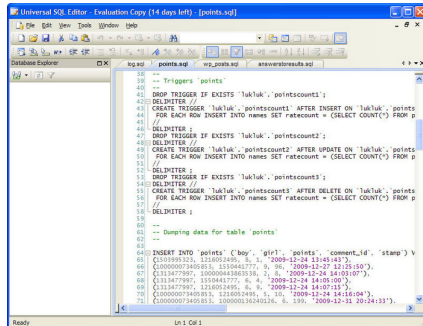


ESTADO DEL ARTE

Spatial SQL

¿Cómo es Spatial SQL?

Es igual que el SQL tradicional:



```
38 --
39 -- Triggers 'points'
40 --
41 DROP TRIGGER IF EXISTS 'lukluk'.pointscount1;
42 DELIMITER //
43 CREATE TRIGGER 'lukluk'.pointscount1 AFTER INSERT ON 'lukluk'.points
44 FOR EACH ROW INSERT INTO names SET ratecount = (SELECT COUNT(*) FROM p
45 //
46 DELIMITER ;
47 DROP TRIGGER IF EXISTS 'lukluk'.pointscount2;
48 DELIMITER //
49 CREATE TRIGGER 'lukluk'.pointscount2 AFTER UPDATE ON 'lukluk'.points
50 FOR EACH ROW INSERT INTO names SET ratecount = (SELECT COUNT(*) FROM p
51 //
52 DELIMITER ;
53 DROP TRIGGER IF EXISTS 'lukluk'.pointscount3;
54 DELIMITER //
55 CREATE TRIGGER 'lukluk'.pointscount3 AFTER DELETE ON 'lukluk'.points
56 FOR EACH ROW INSERT INTO names SET ratecount = (SELECT COUNT(*) FROM p
57 //
58 DELIMITER ;
59 --
60 -- Dumping data for table 'points'
61 --
62 --
63
64 INSERT INTO 'points' ('boy', 'girl', 'points', 'comment_id', 'stamp') V
65 (1303995312, 118052495, 8, 1, '2009-12-24 13:45:43');
66 (100000073405853, 1550441777, 9, 96, '2009-12-27 12:25:50');
67 (1313477997, 10000044580538, 2, 8, '2009-12-24 14:03:07');
68 (1313477997, 1550441777, 6, 4, '2009-12-24 14:05:00');
69 (1313477997, 1216052495, 6, 9, '2009-12-24 14:07:15');
70 (100000073405853, 1216052495, 1, 10, '2009-12-24 14:16:04');
71 (100000073405853, 100000136240126, 6, 139, '2009-12-31 20:24:33');
```

¿Es más complicado?

Puede, pero no tanto como SQL es:

```
roadType,approachSpeed,recordID,Distance(Transform(GeomFromText(POINT(-1.5924443
54.7865551),4326),27700),PointN(Transform(geometry,27700),NumPoints(geometry)) ) AS
    DistanceToEnd,Distance(Transform(GeomFromText(POINT(-1.5924443
54.7865551),4326),27700),PointN(Transform(geometry,27700),1) ) AS
    DistanceFromStart,Distance(Transform(GeomFromText(POINT(-1.5924443
54.7865551),4326),27700),Transform(geometry,27700)) AS Distance FROM roads WHERE
Distance(GeomFromText(POINT(-1.5924443 54.7865551),4326),geometry) < 0.0002 ORDER BY
    Distance(GeomFromText(POINT(-1.5924443 54.7865551),4326),geometry) LIMIT 1
```

¿Cómo es la información espacial?

| | A | B | C |
|----|--|--------------------|---------------------|
| 1 | WKT | Lat | Lon |
| 2 | POINT (-84.147260405808282 9.3899839705711656) | 9.3899839705711656 | -84.147260405808282 |
| 3 | POINT (-84.145554520874811 9.3899522153382691) | 9.3899522153382691 | -84.145554520874811 |
| 4 | POINT (-84.145972945481134 9.39075668034107) | 9.39075668034107 | -84.145972945481134 |
| 5 | POINT (-84.148204543381524 9.3922703396899436) | 9.3922703396899436 | -84.148204543381524 |
| 6 | POINT (-84.147453524857355 9.3926196447542338) | 9.3926196447542338 | -84.147453524857355 |
| 7 | POINT (-84.1489555619057 9.3910530617133077) | 9.3910530617133077 | -84.1489555619057 |
| 8 | POINT (-84.149084307938409 9.3912435924615778) | 9.3912435924615778 | -84.149084307938409 |
| 9 | POINT (-84.146874167710138 9.39211156454428) | 9.39211156454428 | -84.146874167710138 |
| 10 | POINT (-84.1512247107323 9.3940962486339) | 9.3940962486339 | -84.1512247107323 |

Nuevos tipos de datos

- ▶ Point
- ▶ LineString
- ▶ Polygon
- ▶ MultiPoint
- ▶ etcétera...

Relaciones

- ▶ ST_Contains
- ▶ ST_Equals
- ▶ ST_Intersects
- ▶ ST_Touches
- ▶ etcétera...

Constructores

- ▶ ST_MultiLineString
- ▶ ST_Point
- ▶ ST_MultiPolygon
- ▶ ST_MultiCurve
- ▶ etcétera...

Spatial SQL: Ejemplo

Crear una tabla nueva:

```
CREATE TABLE Rio(  
    Nombre varchar(30),  
    Origen varchar(30),  
    Longitud number,  
    Forma LineString );
```

Consulta: encontrar nombres de países que son vecinos de Estados Unidos en la tabla Países.

```
SELECT P1.Nombre  
FROM Países P1, Países P2  
WHERE Touch(P1.Forma,P2.Forma)=1  
AND P2.Nombre = USA
```


Spatial SQL: Ejemplo

Consulta: encontrar la distancia entre dos puntos, dada una tabla de puntos.

```
SELECT ST_Distance(geometrycolumn,  
    ST_GeomFromText('POINT(1 2)',4326)  
FROM tabladepuntos  
ORDER BY  
ST_Distance(geometrycolumn,  
    ST_GeomFromText('POINT(1 2)',4326) LIMIT 10)
```

Alternativas Espaciales

Algunas de las elecciones que podemos hacer hoy en día en materia de DBMS espaciales son:

Open Source

- ▶ MySQL (actualmente viene con soporte espacial)
- ▶ MongoDB
- ▶ MariaDB
- ▶ PostgreSQL + PostGIS
- ▶ SpatialDB
- ▶ y más ...

Pagos

- ▶ Oracle
- ▶ SQL Server

PostgreSQL + PostGIS

PostgreSQL



PostgreSQL

- ▶ DBMS Open Source
- ▶ Soporte para tablas de gran tamaño y distintos tipos de datos

PostGIS

- ▶ Extensión Espacial a PostgreSQL Open Source
- ▶ Basado en las librerías de C

Geometrías Básicas

- ▶ POINT (x y)
- ▶ LINESTRING (x_1 y_1 , ..., x_n y_n)
- ▶ POLYGON (x_1 y_1 , ..., x_n y_n)

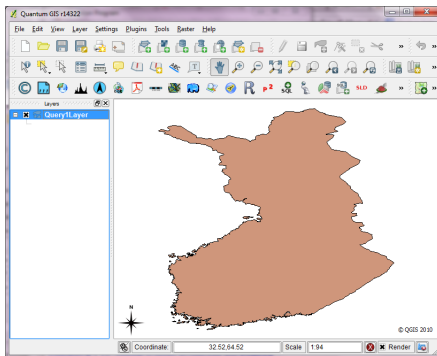
Multi-Geometrías

- ▶ MULTIPOINT ((POINT₁), ..., (POINT _{n}))
- ▶ MULTILINESTRING ((LINESTRING₁, ..., (LINESTRING _{n}))
- ▶ MULTIPOLYGON ((POLYGON₁), ..., (POLYGON _{n}))

PostgreSQL + PostGIS

Los constructores, operadores y relaciones son los que vienen con Spatial SQL, es decir, los mencionados anteriormente.

PostGIS agrega características interesantes como su propio spatial join, GIS overlay functions, primitivas 2D y 3D, además de un entorno de desarrollo amigable.



- ▶ An Introduction to Spatial Database Systems, Güting.
- ▶ MySQL 5.7 Reference Manual.
- ▶ Spatial Database Systems. Design, Implementation and Project Management Series: GeoJournal Library, Yeung A., Hall K.W., Brent G.
- ▶ Tópicos Avanzados de Bases de Datos, Bender C. y Deco C.
- ▶ PostGIS Documentation.