Capítulo 9

Sistemas de Gestión de Bases de datos y SIG

Un Sistema de Gestión de Bases de Datos (SGBD[1]) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a los mismos. Esta definición es prácticamente idéntica a la que se dió anteriormente de Sistema de Información, de hecho normalmente en el nucelo de un SI se sitúa un SGBD. El caso de lo SIG es un poco diferente ya que en principio las bases de datos espaciales no son adecuadas para su manejo con SGBD tradicionales.

Sin embargo, a lo largo del desarrollo de las tecnologías ligadas a los SIG desde los setenta hasta la actualidad, una de las tendencias más claras es el papel, cada vez más importante, que tiene el uso de SGBD para la gestión de datos temáticos como apoyo al SIG. En principio se utilizaron para almacenar los atributos temáticos asociados a un conjunto de entidades espaciales almacenadas en formato vectorial, hoy en día se están empezando a utilizar además para el almacenamiento de la información geométrica (conjunto de coordenadas) de las entidades espaciales. Aunque se han hecho algunos intentos para almacenar información en formato raster en un SGBD, esta opción no resulta eficiente.

9.1. Características fundamentales de un Sistema de Gestión de Base de Datos (SGBD)

Un SGBD permite el almacenamiento, manipulación y consulta de datos pertenecientes a una base de datos organizada en uno o varios ficheros. En el modelo más extendido (base de datos relacional) la base de datos consiste, de cara al usuario, en un conjunto de tablas entre las que se establecen relaciones. A pesar de sus semejanzas (ambos manejan conjuntos de tablas) existen una serie de diferencias fundamentales entre un SGBD y un programa de hoja de cálculo, la principal es que un SGBD permite:

- El método de almacenamiento y el programa que gestiona los datos (**servidor**) son independientes del programa desde el que se lanzan las consultas (**cliente**) (figura **??**).

[1] En inglés DBMS (Data Base Management System)

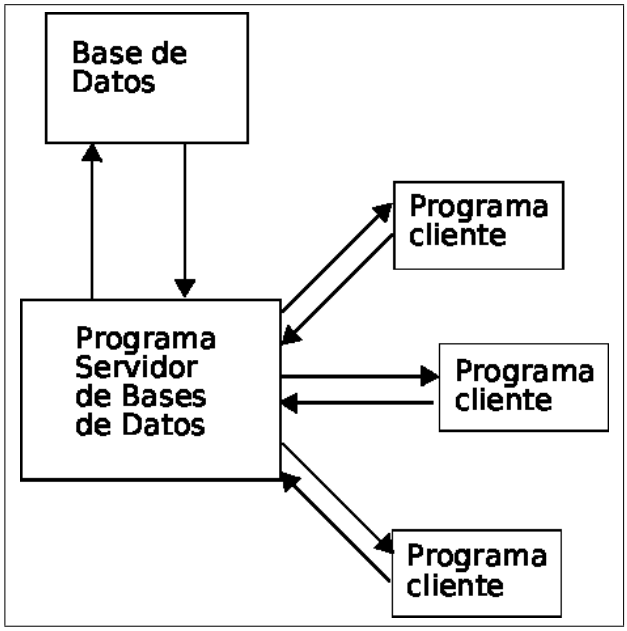

Figura 9.1: Esquema cliente-servidor en una base de datos

- En lugar de primarse la visualización de toda la información, el ojetivo fundamental es permitir **consultas complejas**, cuya resolución está optimizada, expresadas mediante un lenguaje formal.
- El **almacenamiento** de los datos se hace **de forma eficiente aunque oculta** para el usuario y normalmente tiene, al contrario de lo que ocurre con las hojas de cálculo, poco que ver con la estructura con la que los datos se presentan al usuario.
- El acceso concurrente de **múltiples usuarios** autorizados a los datos, realizando operaciones de actualización y consulta de los mismos garantizando la ausencia de problemas de seguridad (debidos a accesos no autorizados) o integridad (pérdida de datos por el intento de varios usuarios de acceder al mismo fichero al mismo tiempo.

El programa servidor suele activarse al arrancar el ordenador, podría compararse a un bibliotecario que recibe peticiones (consultas) de diferentes programas clientes de base de datos, consulta la base de datos y entrega al cliente el resultado de la consulta realizada. Si dos usuarios solicitan al mismo tiempo una modificación de los datos, el programa servidor se encarga de hacerlas ordenadamente para evitar perder datos (lo que ocurriría si ambos usuarios abrieran y modificaran a la vez un fichero con la base de datos.

El diseño de una base de datos implica codificar en formato digital ciertos aspectos del mundo real. Esta codificación implica los mismos 3 pasos que ya se mencionaron en el tema 2, es decir:

- Modelo conceptual

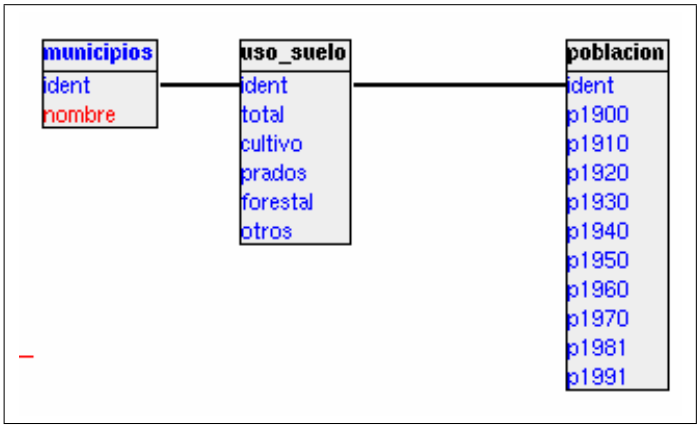

Figura 9.2: Esquema de base de datos relacional

- Modelo lógico
- Modelo digital o implementación física de la base de datos

Hoy en día existen dos grandes modelos, las **bases de datos relacionales** y el **modelo orientado a objetos (modelo OO)**, y un modelo híbrido denominado **modelo Objeto-Relacional (modelo OR)**. En cualquier manual de bases de datos puede encontrarse información acerca de modelos más antiguos.

9.2. Bases de datos relacionales

Es el modelo más utilizado hoy en día. Una base de datos relacional es básicamente un conjunto de tablas, similares a las tablas de una hoja de cálculo, formadas por filas (registros) y columnas (campos). Los registros representan cada uno de los objetos descritos en la tabla y los campos los atributos (variables de cualquier tipo) de los objetos. En el modelo relacional de base de datos, las tablas comparten algún campo entre ellas. Estos campos compartidos van a servir para establecer relaciones entre las tablas que permitan consultas complejas (figura **??**). En esta figura aparecen tres tablas con información municipal, en la primera aparecen los nombres de los municipios, en la segunda el porcentaje en cada municipio de los diferentes usos del suelo y en la tercera la población en cada municipio lo largo del siglo XX. Como campo común aparece **ident**, se trata de un identificador numérico, único para cada municipio[2]

La idea básica de las bases de datos relacionales es la existencia de *entidades* (filas en una tabla) caracterizadas por *atributos* (columnas en la tabla). Cada tabla almacena entidades del mismo tipo y entre entidades de distinto

[2]Es preferible utilizar valores numéricos en lugar de una cadena de caracteres ya que se ahorra espacio y se evitan problemas con el uso de mayúsculas, acentos, etc.

tipo se establecen *relaciones*[3]. Las tablas comparten algún campo entre ellas, estos campos compartidos van a servir para establecer relaciones entre las tablas. Los atributos pueden ser de unos pocos tipos simples:

- Números enteros
- Números reales
- Cadena de caracteres de longitud variable

Estos tipos simples se denominan *tipos atómicos* y permiten una mayor eficacia en el manejo de la base de datos pero a costa de reducir la flexibilidad a la hora de manejar los elementos complejos del mundo real y dificultar la gestión de datos espaciales, en general suponen un problema para cualquier tipo de datos geométricos.

Las relaciones que se establecen entre los diferentes elementos de dos tablas en una base de datos relacional pueden ser de tres tipos distintos:

- **Relaciones uno a uno**, se establecen entre una entidad de una tabla y otra entidad de otra tabla. Un ejemplo aparece en la figura **??**.
- **Relaciones uno a varios**, se establecen entre varias entidades de una tabla y una entidad de otra tabla. Un ejemplo sería una tabla de pluviómetros en la que se indicara el municipio en el que se encuentra. La relación sería entre un municipio y varios pluviómetros
- **Relaciones varios a varios**, se establecen entre varias entidades de cada una de las tablas. Un ejemplo sería una tabla con retenes de bomberos y otra con espacios naturales a los que cada uno debe acudir en caso de incendio.

9.2.1. SQL. El lenguaje de consultas para las bases de datos relacionales

El lenguaje de consultas **SQL** (Lenguaje Estructurado de Consultas) se ha convertido, debido a su eficiencia, en un estandar para las bases de datos relacionales. A pesar de su estandarización se han desarrollado, sobre una base común, diversas versiones ampliadas como las de Oracle o la de Microsoft SQL server.

Es un **lenguaje declarativo** en el que las órdenes especifican cual debe ser el resultado y no la manera de conseguirlo (como ocurre en los **lenguajes procedimentales**). Al ser declarativo es muy sistemático, sencillo y con una curva de aprendizaje muy agradable. Sin embargo los lenguajes declarativos carecen de la potencia de los procedimentales. El gran éxito de las bases de datos relacionales se debe en parte a la posibilidad de usar este lenguaje. Incluye diversos tipos de capacidades:

- Comandos para la **definición y creación** de una base de datos (CREATE TABLE).
- Comandos para **inserción, borrado o modificación** de datos (INSERT, DELETE, UPDATE).

[3]En la bibliografía inglesa sobre bases de datos se habla de *relations* (tablas) y *relationships* relaciones entre las tablas. El término base de datos relacional hace en realidad referencia a la organización de los datos en forma de tablas, no a las relaziones entre ellas

- Comandos para la **consulta** de datos seleccionados de acuerdo a criterios complejos que involucran diversas tablas relacionadas por un campo común (SELECT).
- Capacidades aritméticas: En SQL es posible incluir operaciones aritméticas así como comparaciones, por ejemplo $A > B + 3$.
- Funciones matemáticas ($sqrt(x)$, $cos(x)$) o de manejo de textos.
- **Asignación y comandos de impresión**: es posible imprimir una tabla construida por una consulta o almacenarla como una nueva tabla.
- **Funciones agregadas**: Operaciones tales como promedio (avg), desviación típica (stddev), suma (sum), máximo (max), etc. se pueden aplicar a las columnas de una tabla para obtener una cantidad única y, a su vez, incluirla en consultas más complejas.

En una base de datos relacional, los resultados de la consulta van a ser datos individuales, tuplas[4] o tablas generados a partir de consultas en las que se establecen una serie de condiciones basadas en valores numéricos. Por ejemplo una típica consulta sobre una tabla en una base de datos relacional, utilizando SQL podría ser:

```
SELECT id, nombre, pob1991
FROM municipios
WHERE pob1991>20000;
```

el resultado será una tabla en la que tendremos tres columnas (id, nombre, poblacion) procedentes de la tabla municipios, las filas corresponderán sólo a aquellos casos en los que la poblacion en 1991 (columna pob1991) sea mayor que 20000. En el caso de que sólo uno de los municipios cumpliera la condición obtendríamos una sola fila (una tupla) y en caso de que la consulta fuera:

```
SELECT pob1991
FROM municipios
WHERE pob1991>20000;
```

obtendríamos un sólo número, la población del municipio más poblado.

9.2.2. SIG y bases de datos relacionales: El modelo geo-relacional

Lo más habitual es utilizar el SGBD para almacenar la información temática y el SIG para la información geométrica y topológica. Una de las funcionalidades de este modelo será el enlazado de ambos tipos de información que se almacenana de formas completamente diferentes. Se trata del modelo de datos *geo-relacional*.

[4] una tupla equivale a una fila en una tabla

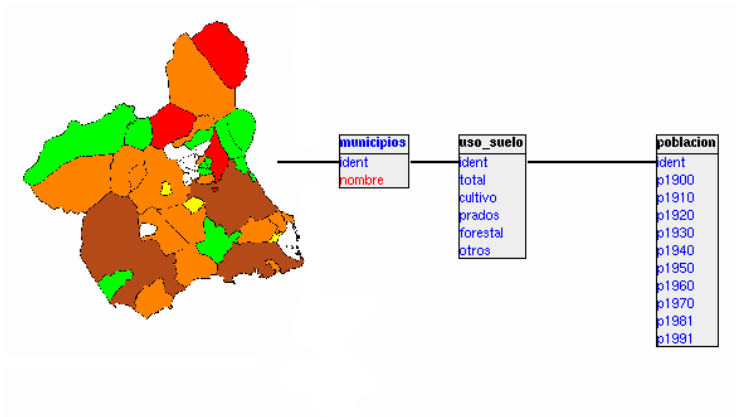

Figura 9.3: Esquema de base de datos geo-relacional

El mayor interés del modelo geo-relacional estará en poder lanzar una consulta SQL y obtener una o varias entidades espacial (en lugar de número, tabla o fila) como respuesta. Para ello debe enlazarse la base de datos espacial (mapa vectorial) con la base de datos temática (tablas) mediante una columna en una de las tablas de la base de datos que contenga los mismos identificadores que las entidades en la base de datos espacial.

Podemos pensar en un mapa vectorial como en una tabla en la que cada registro (fila) es un objeto (polígono, linea o punto) que contiene un campo identificador y un campo que contiene la localización (conjunto de coordenadas X e Y de tamaño, lógicamente, variable). El hecho de que esta información se presente en forma de tabla o en forma de mapa es simplemente una cuestión de conveniencia.

Si pedimos, como resultados de una consulta a la base de datos temática, estos identificadores comunes, en realidad lo que estamos obteniendo son objetos espaciales (polígonos en el caso de los municipios). Los resultados de las consultas podrían presentarse de esta manera en forma de mapa en lugar de en forma de tabla de modo que a los diferentes polígonos se le asignarían diferentes colores en función de que se cumpliera o no una condición, o de los valores que adoptase una variable o índice. Por ejemplo la consulta

```
SELECT ident, nombre
FROM municipios
WHERE 1000*(pob1991-pob1981)/pob1981>0 AND pob1981>0;
```

para obtener aquellos municipios con una tasa de crecimiento de población positiva entre 1981 y 1991 en tantos por mil, podría representarse en un SIG tal como aparece en la figura **??**.

Una consulta similar a la anterior pero estableciendo una reclasificación por colores daría el resultado que puede verse en la figura **??** en la que el que el color rojo indica valores mayores de 50, el amarillo entre 30 y 50, el verde entre 20 y 30, el azul entre 10 y 20 y el blanco menor de 10.

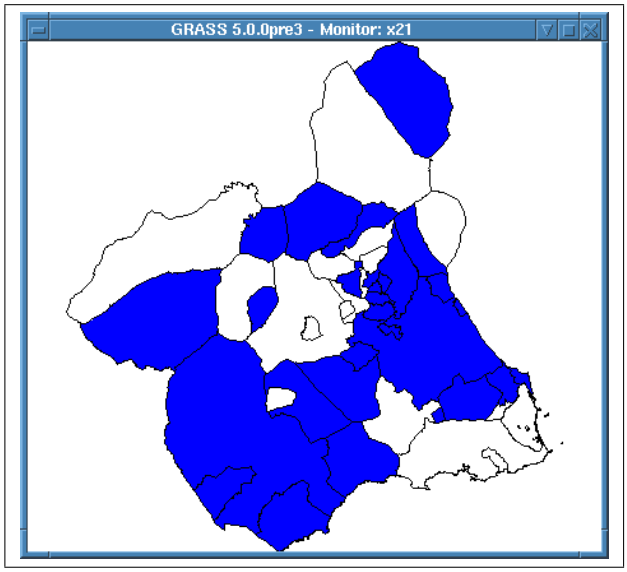

Figura 9.4: Municipios con crecimiento de población positivo entre 1981 y 1991

En estos casos se necesita un módulo específico que transforme los resultados de las consultas en una serie de reglas para pintar los polígonos asignando al mismo tiempo una paleta de colores definida por el usuario.

En definitiva la única diferencia entre el trabajo de un gestor tradicional de bases de datos y el enlace de un SIG a base de datos es el modo de presentación (tabla o mapa). Casi todo el trabajo lo hace el gestor de bases de datos y el Sistema de Información Geográfica, se limita a presentar los resultados.

Hasta ahora lo que hemos hecho es obtener objetos espaciales como resultado de una consulta, pero cuando se trabaja con un SIG enlazado a una base de datos, se pretende que las consultas incluyan tambien condiciones espaciales. Incluso deberíamos ser capaces de llevar a cabo consultas interactivas en las que las condiciones se formulan en función de donde haya pinchado el usuario en un mapa mostrado en pantalla.

Sin embargo en el modelo geo-relacional toda la información geométrica y topológica está en el SIG no en el SGBD por tanto las consultas deberán **preprocesarse** y **postprocesarse**.

Preprocesamiento significa que el módulo encargado de construir de forma automática consultas SQL como las que hemos visto antes, y lanzarlas al programa servidor de bases de datos, deberá hacerlo teniendo en cuenta una serie de criterios espaciales definidos por el usuario. Por ejemplo, si el usuario pincha en la pantalla dentro de un polígono esperando obtener nombre y población del municipio, el módulo deberá determinar de que polígono se trata e incluir su identificador, por ejemplo 17, como condición que debe cumplirse:

```
SELECT nombre, pob1991
FROM municipios
WHERE id==17;
```

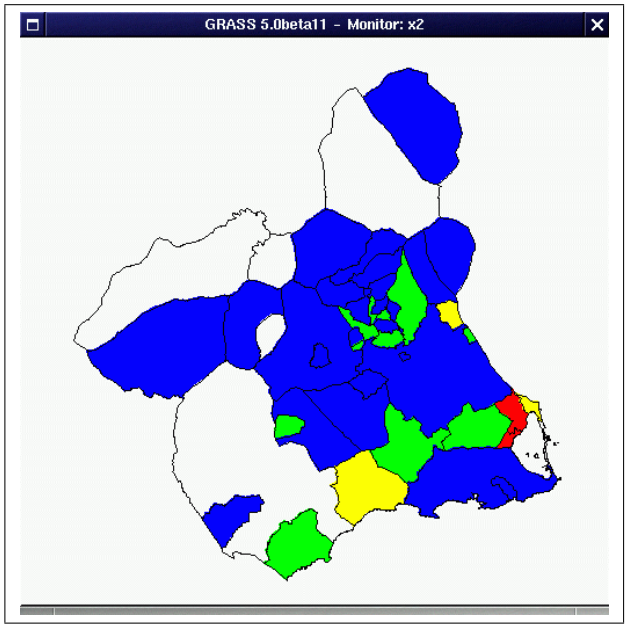

Figura 9.5: Crecimiento de población entre 1981 y 1991

Postprocesamiento implica que los resultados de la consulta SQL deberán filtrarse para determinar cuales cumplen determinadas condiciones relacionada con el espacio. Para ello, una de las columnas pedidas en la consulta ha de ser el identificador a partir del cual se obtiene, ya en el SIG, la geometría del polígono a la que se puede aplicar la operación de análisis espacial (distancia, cruce, inclusión, adyacencia, etc.) necesaria para derminar si se cumple o no la condición. Aquellos casos en los que si se cumple constituye la salida del módulo, el resto se deshechan.

9.3. Bases de datos orientadas a objetos

El modelo OO da lugar a las **bases de datos orientados a objetos**. Es un concepto totalmente distinto al de las bases de datos relacionales que responde al paradigma de la orientación a objetos desarrollado en programación de ordenadores en los últimos años.

Al no estar constreñido por el formato de tablas, cuyas columnas responden a tipos atómicos, permite una mayor flexibilidad a la hora de incorporar tipos más complejos como los tipos geométricos (puntos, lineas, polígonos, etc.) por tanto es un modelo, *a priori*, más adecuado para el trabajo con un Sistema de Información Geográfica.

Se parte del concepto de **clase** que agrupa a todos los objetos que comparten una serie de **atributos**, estos atributos pueden incluir la geometría del objeto, las relaciones topológicas y propiedades temáticas. Junto con los

atributos, las clases incluyen un conjunto de **métodos** (acciones que pueden llevarse a cabo sobre los objetos). No se permite el acceso directo a los atributos sino sólo mediante sus métodos, esta propiedad se denomina **encapsulamiento** e incrementa la seguridad de los datos ante errores. Otra característica interesante es la **herencia** por la cual unos objetos pueden derivar de otros heredando sus atributos y métodos e incorporando otros.

Por ejemplo podría definirse la clase *polígono* incluyendo como atributos el *área* y el *perímetro* y como métodos el *cálculo del área* y el *cálculo del perímetro*. Posteriormente podría crearse la clase *municipio* que hereda los atributos y métodos de su clase padre (polígono) incorporando una serie de nuevos atributos (*población*, *renta per cápita*, etc.) y métodos como por ejemplo el *cálculo de la densidad de población* que se ejecuta dividiendo el atributo *población* entre el atributo *area*. Por otro lado podemos crear la clase *cuenca fluvial* que hereda atributos y métodos de la clase *polígono* y define atributos propios como puede ser *río al que desemboca* y métodos como *cálculo de caudal pico*.

Por su complejidad, las bases de datos orientadas a objetos no utilizan SQL e incluyen un lenguaje específico para hacer las consultas.

Las bases de datos orientadas a objetos no han tenido, sin embargo, un gran desarrollo, al menos hasta el momento. Entre las causas de este hecho cabe destacar el éxito de SQL y su tremenda eficiencia y el carácter altamente intuitivo de las tablas del modelo relacional. Por ello, se ha desarrollado un modelo híbrido que trata de capturar lo esencial de la orientación a objetos sin perder la eficiencia del modelo relacional. Se trata de las bases de datos objeto-relacionales.

9.4. Bases de datos objeto-relacionales

La idea es mantener el esquema de tablas entre las que se establecen relaciones pero permitiendo como atributos, además de los tipos atómicos, tipos más complejos denominados **tipos abstractos de datos (ADT)** que admiten objetos geométricos. Para ello el SGBD debe modificarse para admitir nuevas capacidades:

- Deben poder definirse nuevos tipos de datos que permitan almacenar la geometría (puntos, lineas, polígonos, etc.).
- Las funciones y operadores ya existentes se adaptan a estos datos espaciales.
- El lenguaje SQL se extiende para manipular datos espaciales, incluyendo funciones como distancia, cruce de lineas, punto en polígono, etc., que se vieron el el tema dedicado al formato vectorial.
- En el nivel físico, es decir en el modelo digital, se requieren cambios profundos.

Hasta el año 2000 aproximadamente el modelo geo-relacional era casi la única opción para trabajar con SIG enlazados a bases de datos, ultimamente se tiende a adoptar la segunda, en parte como resultado de la entrada de las empresas de desarrollo de bases de datos en el mercado de los SIG. Entre las ventajas que aporta este modelo destaca que se gana en velocidad al evitar gran parte del procesamiento en SIG y se permite que diversos programas cliente puedan acceder de forma concurrente al programa servidor.

El inconveniente es que las extensiones de SQL para incluir operadores espaciales se hacen demasiado complejas. Por otro lado sigue siendo necesaria la existencia de herramientas de SIG que lean la información almacenada en la base de datos y la muestren en pantalla. Puedes consultar las especificaciones para SQL del OpenGIS Consortium para obtener más información al respecto.

9.4.1. Concepto de Geodatabase

El concepto de **Geodatabase** es uno de los que han experimentado en los últimos años una mayor expansión en el mundo de los SIG. Se trata simplemente de una base de datos que almacena toda la información relativa a un conjunto de entidades espaciales (geometría, topología, identificadores, datos temáticos, etc.). Las ventajas de este modelo de trabajo son varías:

- Posibilidad de usar SQL, una versión ampliada de SQL en realidad, para hacer consultas y análisis sobre mapas vectoriales
- Mayor integración, en una sóla herramienta, de todas las funciones para trabajar con información vectorial

El inconveniente es que se necesita un programa externo, el SIG de toda la vida, para acceder a los datos y visualizarlos.

Entre los programas que permiten trabajar con geodatabases cabe destacar 2, en primer lugar Oracle spatial y en segundo lugar PostgreSQL + PostGIS. Oracle está considerado como el mejor programa de gestión de base de datos, siendo uno de sus inconvenientes su elevado precio. PostgreSQL es una alternativa libre (y gratuita) que realmete no desmerece apenas de Oracle. PostGIS es una extensión, también libre, de PostgreSQL que le permite trabajar con geodatabases.

A continuación se exponen algunos de los operadores y funciones que permiten trabajar con diferentes tipos de entidades en una Geodatabase. Se han utilizado nombres en castellano que, obviamente, no corresponden con las órdenes reales de ningún programa pero se basan en las órdenes de PostGIS. Las figuras **??** y **??** permiten visualizar que es lo que comprueban o calculan algunas de las órdenes.

Operadores que devuelven cierto o falso (figuras ??)

- es_igual(entidad,entidad)
- se_solapa_con(entidad,entidad)
- toca_a(entidad,entidad)
- cruza(linea,entidad)
- está_dentro_de(entidad,polígono)

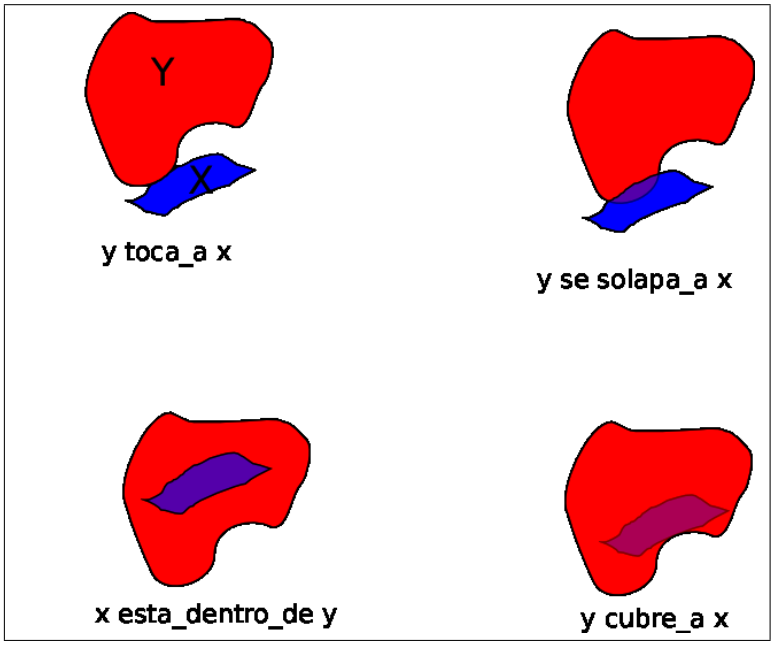

Figura 9.6: Operadores que devuelven cierto o falso

- cubre_a(polígono,entidad)
- está_relacionada(entidad,entidad,matriz_de_relaciones)

Funciones que devuelven una geometría (figuras ?? y ??)

- buffer(entidad,distancia)
- convexhull(puntos) Polígono cuyos vértices coinciden con algunos de los puntos de una muestra de puntos y que contiene todos los demás, cuyos ángulos son todos menores de 180^o vistos desde dentro del polígono (figura **??**).
- Intersección(polígono,entidad)
- Union(entidad, entidad) Equivalente a una suma
- Diferencia(entidad,entidad)

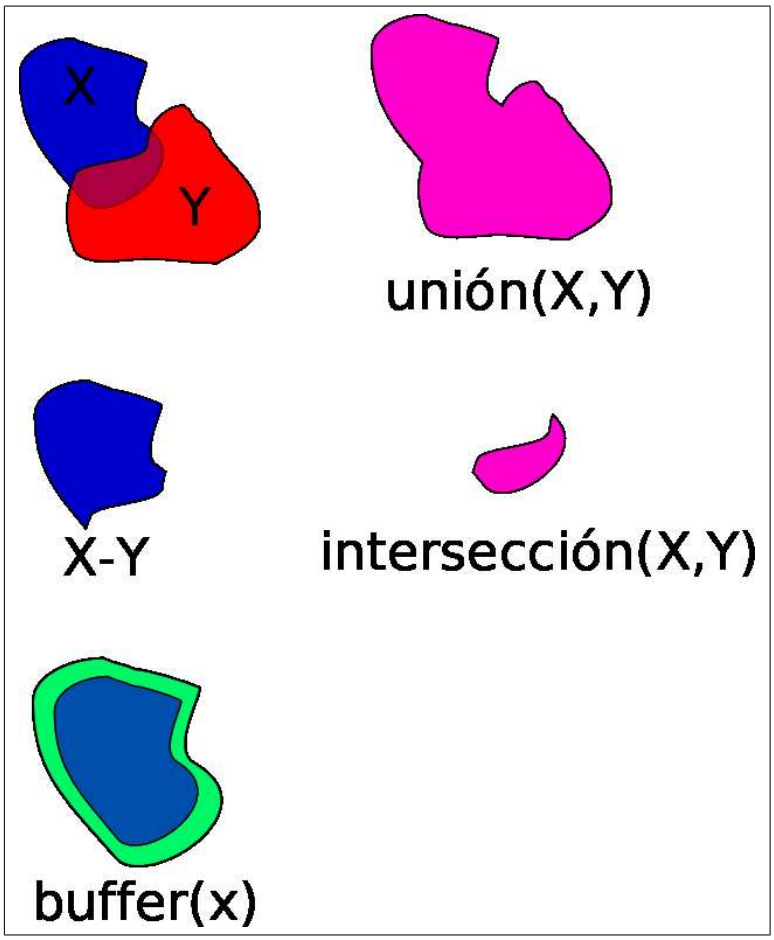

Figura 9.7: Funciones que devuelven una geometría

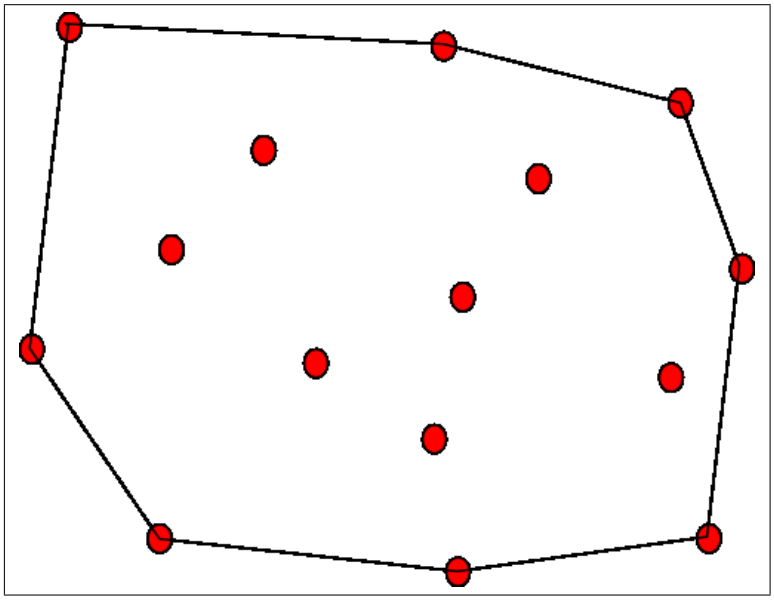

Figura 9.8: Conjunto convexo formado a partir de una muestra de puntos

Funciones que devuelven números

- X(entidad)
- Y(entidad)
- Z(entidad)
- Longitud(entidad)
- Area(polígono)
- NumPoints(entidad)
- Distancia(entidad,entidad)

Las funciones x, Y y Z devuelven las coordenadas X, Y y Z (repectivamente) de todos los vértices de la entidad que se le pasa. Las funciones Longitud y Area devuelven estas magnitudes (si se trata de un polígono, Distancia devolverá el perímetro). NumPoints devuelve el número de vértices y Distancia la distancia mínima entre dos vértices que pertenecen cada uno a cada una de las entidades que se pasan a la función.

Funciones que devuelven un punto:

- Nodo_inicial(linea), devuelve el primer nodo de la linea

- Nodo_final(linea), devuelve el primer nodo de la linea
- Centroide(entidad), devuelve un punto situado en el centro geométrico del objeto que se pasa (se calcula como la media de todas las coordenadas X de los vértices y la media de todas las coordenadas Y de los vértices.

9.5. Consultas SQL con capas raster y de puntos

Aunque generalmente se asume que el enlace de un SIG con una base de datos relacional incumbe fundamentalemente al formato vectorial, nada impide enlazar una base de datos con un mapa raster que contenga polígonos o una variable cualitativa.

Por ejemplo si se tiene un mapa raster que contiene tipos de suelo y una base de datos en la que a cada tipo de suelo se asocian diversas propiedades edáficas, pueden utilizarse los resultados de consultas SQL para transformar, mediante reclasificación, el mapa de suelos en diversos mapas de variables edáficas.

En el caso de los mapas de puntos, las propiedades geométricas son mínimas (dos coordenadas) y las topológicas inexistentes. Por tanto los mapas de puntos pueden almacenarse sin problemas como tablas en una base de datos. Las consultas permitirán obtener tripletes X,Y,Z que pueden utilizarse como información de entrada para diversas herramientas SIG como puede ser la interpolación.

9.6. Bibliografía

- desarrolloweb.com *Tutorial de SQL* http://www.desarrolloweb.com/manuales/9/
- OpenGIS Consortium (1999) *OpenGIS Simple Features Specification For SQL* http://www.opengis.org/docs/99-049.pdf
- Rigaux,P.; Scholl,M. & Voisard,A. (2001) *Introduction to Spatial Databases: Applications to GIS*, Morgan Kaufmann, 400 pp.
- Shekhar,S. & Chawla,S. (2002) *Spatial Databases: A Tour*, Prentice Hall ,300 pp.