

Bases de Datos Geográficas I



Taller de Sistemas de Información Geográfica



Open Geospatial Consortium (OGC)

- ❑ El Open Geospatial Consortium (OGC) es un consorcio internacional que agrupa empresa, universidades y organismos estatales (alrededor de 400) con las siguientes metas fundamentales:
 - Proveer de estándares abiertos, gratuitos y públicos.
 - Liderar la creación de estándares que permitan que el contenido y los servicios geo-espaciales se integren a procesos cívicos y de negocios, la Web Espacial y los Sistemas Empresariales
 - Facilitar la adopción de arquitecturas de referencia abiertas en materia de información espacial.



OGC Simple Features Standard (SFS)

- ❑ OGC define un modelo de datos para objetos espaciales conocido como Simple Features Standard (SFS)(ISO 19125:2004).
- ❑ El estándar se divide en dos partes:
 1. Arquitectura común (Modelo de Objetos)
 2. Implementación en SQL (Permite construir bases de datos geográficas)





Clases de Objetos Geográficos

□ Geometry (abstracta)

- Es la clase base de la jerarquía.
- Permite representar geometrías planas (hasta 2 dimensiones) en espacios de coordenadas R2, R3 y R4.
 - En R2 se define cada punto por coordenadas (x,y)
 - En R3 se define cada punto por coordenadas (x,y,z) o (x,y,m).
 - En R4 se define cada punto por coordenadas (x,y,z,m)
 - La coordenada “z” representa altitud sobre el nivel del mar.
 - La coordenada “m” (measure) representa una medida que depende de las demás coordenadas. Ej.: en una geometría que representa una red fluvial, el valor “m” de cada punto puede ser la distancia a la desembocadura del río al que pertenece.



Clases de Objetos Geográficos

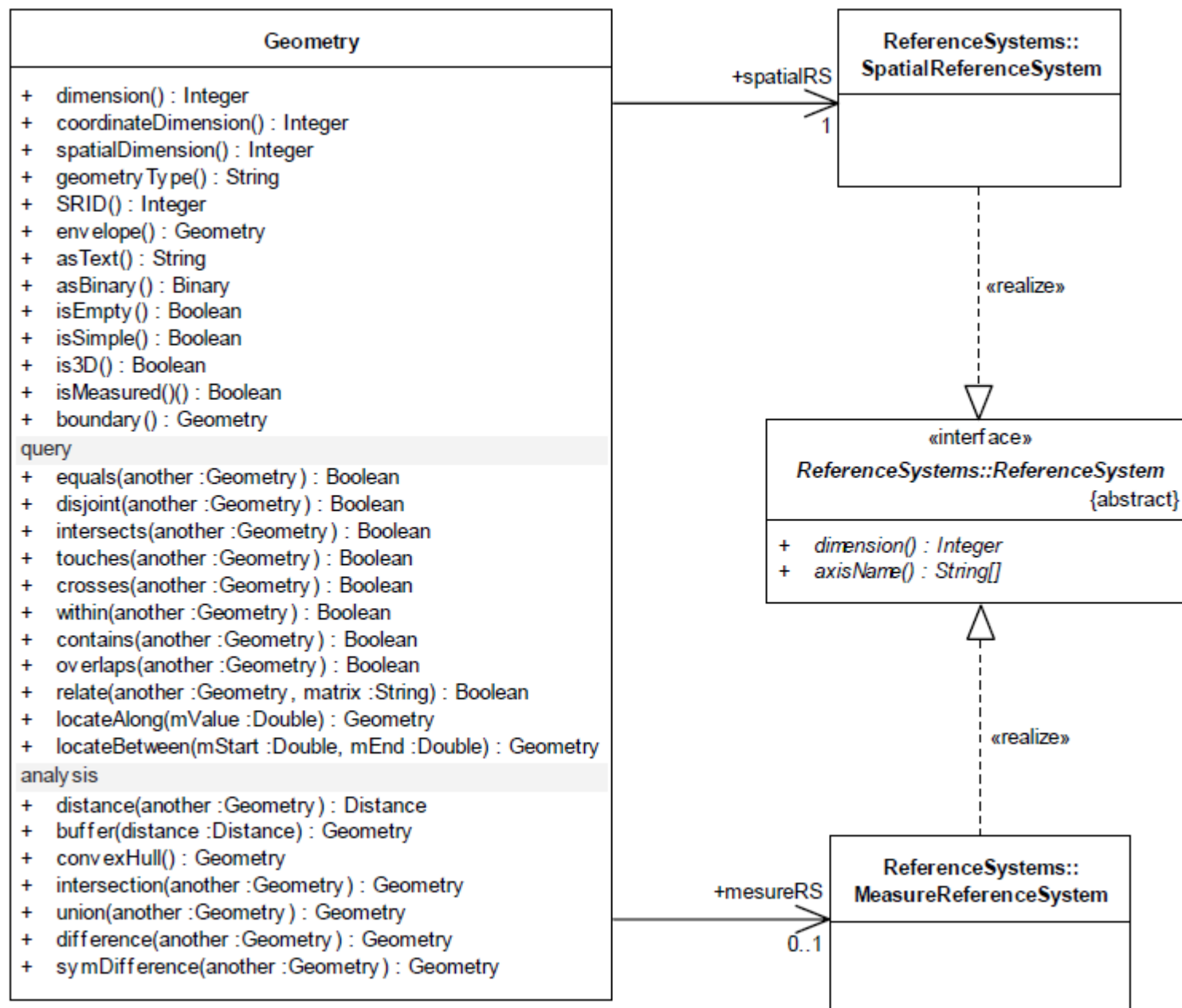
- o Métodos básicos:
 - *dimension(): La dimensión inherente a la geometría. Devuelve*
 - 1 si es la geometría vacía
 - 0 si longitud=área=0
 - 1 si longitud $\neq$ 0, área=0
 - 2 si longitud $\neq$ 0, área $\neq$ 0
 - *geometryType(): Devuelve el subtipo concreto al que pertenece el objeto (ej. Polygon, Point, etc.)*
 - *SRID(): Devuelve el ID del Sistema Espacial de Referencia utilizado para georreferenciar este objeto.*



Clases de Objetos Geográficos

- *asText(): Devuelve la representación WKT del objeto.*
- *asBinary(): Devuelve la representación WKB del objeto.*
- *isSimple(): Devuelve 1 si el objeto representa una geometría sin propiedades anómalas, como autointersecciones o autotangencia.*
- *isEmpty(): Devuelve 1 si el objeto representa la geom. vacía (ningún punto). El conjunto vacío es simple.*
- *is3D(): Devuelve 1 si el objeto tiene una coordenada z.*
- *isMeasured(): Devuelve 1 si el objeto tiene una coordenada m.*
- *boundary(): Devuelve otra geometría con la frontera de este objeto.*
- *envelope(): Devuelve el mínimo rectángulo delimitador (minimum bounding box, MBR) que contiene a la geometría.*





Clases de Objetos Geográficos

□ Point

- Objeto geométrico 0D (punto).
- Coordenadas x,y (z,m opcionales).
- Su frontera es el conjunto vacío.
- Su envelope es el propio punto.
- Posee métodos para obtener sus coordenadas



Clases de Objetos Geográficos

□ Curve (abstracta)

- Objeto geométrico 1D que representa la curva definida por una secuencia de puntos y la interpolación entre los mismos.
- Métodos:
 - *length(): el largo en su SRS asociado*
 - *startPoint(), endPoint(): punto inicial y final*
 - *IsClosed(): devuelve 1 si startPoint=endPoint*
 - *IsRing(): devuelve 1 si IsClosed()=1 y no pasa mas de una vez por el mismo punto (es simple).*
- La frontera de una curva cerrada es el conjunto vacío.
- La frontera de una curva abierta son el startPoint y endPoint.



Clases de Objetos Geográficos

❑ LineString

- Es una curva con interpolación lineal entre los puntos.
- Su envelope es un polígono, o si es una línea vertical u horizontal, es la propia línea.
- Métodos:
 - *numPoints()*: devuelve cantidad de puntos
 - *pointN()*: devuelve el punto N

❑ Line

- Es un LineString de 2 puntos (un segmento de recta)

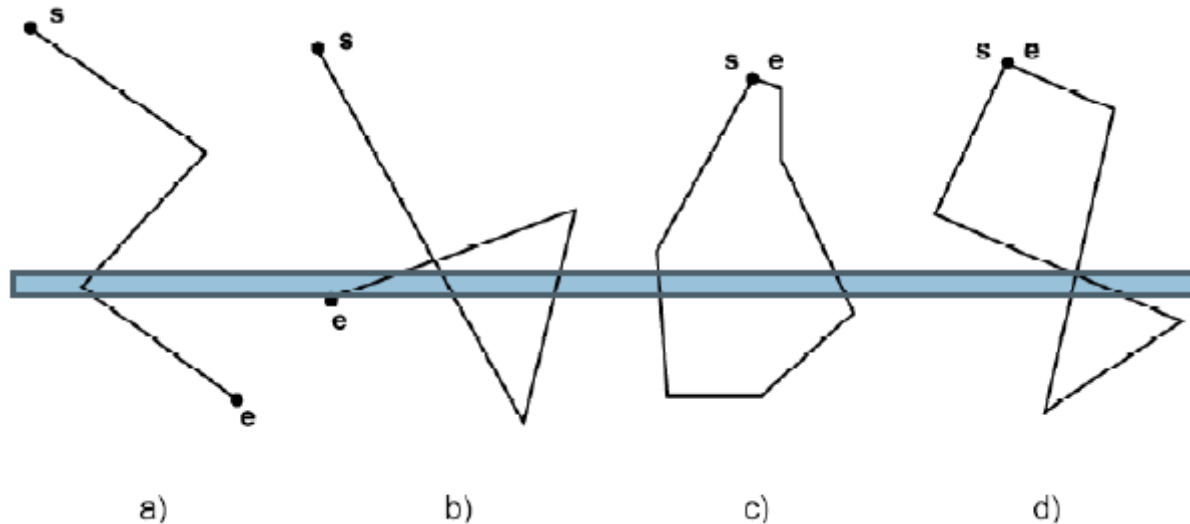
❑ LinearRing

- En un LineString simple y cerrado (un anillo)



Clases de Objetos Geográficos

□ Diferentes tipos de LineString



Clases de Objetos Geográficos

❑ Surface (abstracta)

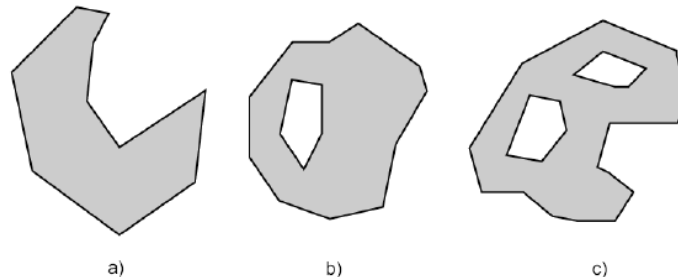
- Objeto geométrico 2D que representa una superficie.
- Métodos:
 - *area(): devuelve el área en el SRS correspondiente.*
 - *centroid(): devuelve el centroide (baricentro) de la superficie (puede ser un punto exterior)*
 - *pointOnSurface(): devuelve un punto cualquiera perteneciente a la superficie.*



Clases de Objetos Geográficos

□ Polygon

- Es una superficie plana definida por un LinearRing exterior y 0 o más LinearRings interiores, para permitir polígonos con huecos.
- Los polígonos son objetos simples: no existe intersección entre sus contornos.
- Métodos:
 - *exteriorRing(): devuelve el anillo exterior*
 - *numInteriorRings(): devuelve la cantidad de anillos interiores.*
 - *interiorRingN(): devuelve el anillo interior N*
- La frontera de un polígono son su anillo exterior y sus anillos interiores



Clases de Objetos Geográficos

❑ GeometryCollection

- Colección de geometrías de cualquier tipo
- No existe relación implícita entre sus elementos.
- Todos sus elementos deben estar en el mismo SRS.
- Permite trabajar con un conjunto de geometrías como una unidad que tiene atributos no espaciales comunes.
 - *Ej. La red fluvial uruguaya, los territorios del Reino Unido y sus islas, etc.*

❑ MultiPoint

- Colección geométrica de puntos.
- Su frontera es el conjunto vacío.
- Un multipunto es simple si no tiene puntos repetidos.

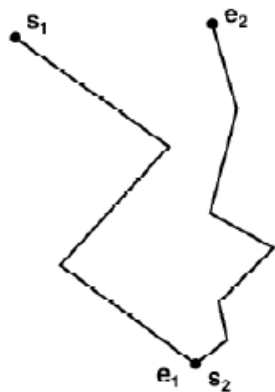


Clases de Objetos Geográficos

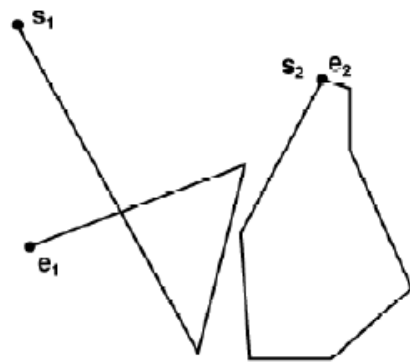
- ❑ MultiCurve (abstracta)
 - La frontera de una multicurva son los puntos que están un número impar de veces en la frontera de sus elementos. (regla de “unión módulo 2”)
- ❑ MultiLineString
 - Es una MultiCurve cuyos elementos son LineStrings.
- ❑ MultiSurface (abstracta)
 - La frontera de una multisuperficie son los puntos que están un número impar de veces en la frontera de sus elementos. (regla de “unión módulo 2”)
- ❑ MultiPolygon
 - Es una MultiSurface cuyos elementos son Polygons.



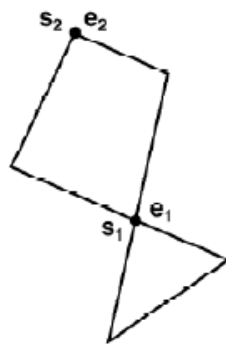
Clases de Objetos Geográficos



a)



b)



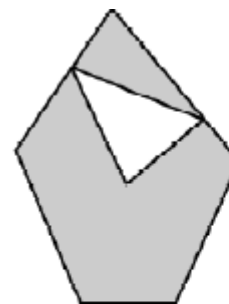
c)



a)



b)



c)



d)



Well-know Binary (WKB)

- Permite representar geometrías mediante un flujo de bytes.
- Ej. un polígono determinado por dos anillos puede representarse de la siguiente manera.

B=1	T=3	NR=2	NP=3	X1	Y1	X2	Y2	X3	Y3	NP=3	X1	Y1	X2	Y2	X3	Y3
-----	-----	------	------	----	----	----	----	----	----	------	----	----	----	----	----	----

B = orden de bytes

T = tipo de figura (polígono)

NR = numero de anillos (2)

NP = numero de puntos (3 para cada anillo)



Well-know Text (WKT)

- ❑ Permite representar geometrías mediante un texto legible por personas.
- ❑ Ej. Punto, Línea, Multilínea, Polígono, Multipolígono en WKT.

```
POINT(2572292.2 5631150.7)
```

```
LINESTRING (2566006.4 5633207.9, 2566028.6 5633215.1, 2566062.3 5633227.1)
```

```
MULTILINESTRING((2566006.4 5633207.9, 2566028.6 5633215.1), (2566062.3  
5633227.1, 2566083 5633234.8))
```

```
POLYGON (2568262.1 5635344.1, 2568298.5 5635387.6, 2568261.04 5635276.15,  
2568262.1 5635344.1);
```

```
MULTIPOLYGON(((2568262.1 5635344.1, 2568298.5 5635387.6, 2568261.04 5635276.15,  
2568262.1 5635344.1), (2568194.2 5635136.4, 2568199.6 5635264.2, 2568200.8  
5635134.7, 2568194.2 5635136.4 )))
```



SFS - SQL

- ❑ Se definen 3 tipos de tablas:
 - FEATURE_TABLE: Es toda tabla que almacena un conjunto de *features* (objetos geográficos). Corresponde al concepto de *layer* (capa geográfica). Cada fila es un objeto geográfico y cada columna es una propiedad de ese objeto. Una de esas columnas debe corresponder a la geometría de ese objeto (o una FK a la misma).
 - Ej. Una capa de polígonos que representan ciudades la representamos como una *feature table* Ciudad(nombre, país, población, geometría, *gid*)



SFS - SQL

- GEOMETRY_TABLE : Es toda tabla que almacena geometrías, en el caso que la *feature table* correspondiente no las almacene directamente. Cada fila posee un identificador geográfico (GID).
- GEOMETRY_COLUMNS: Tabla de metadatos que posee una fila por cada columna geometría de la base, con los siguientes atributos:
 - ID de la *feature table* a la que corresponde la columna geometría
 - El nombre de la columna geometría
 - El SRID de la columna geometría
 - El tipo de geometría (Point, LineString, etc)
 - La dimensión del SRS utilizado (2D, 3D, etc.)
 - El ID de la *geometry_table* que almacena la geometría (podría ser la misma *feature table*)



SFS - SQL

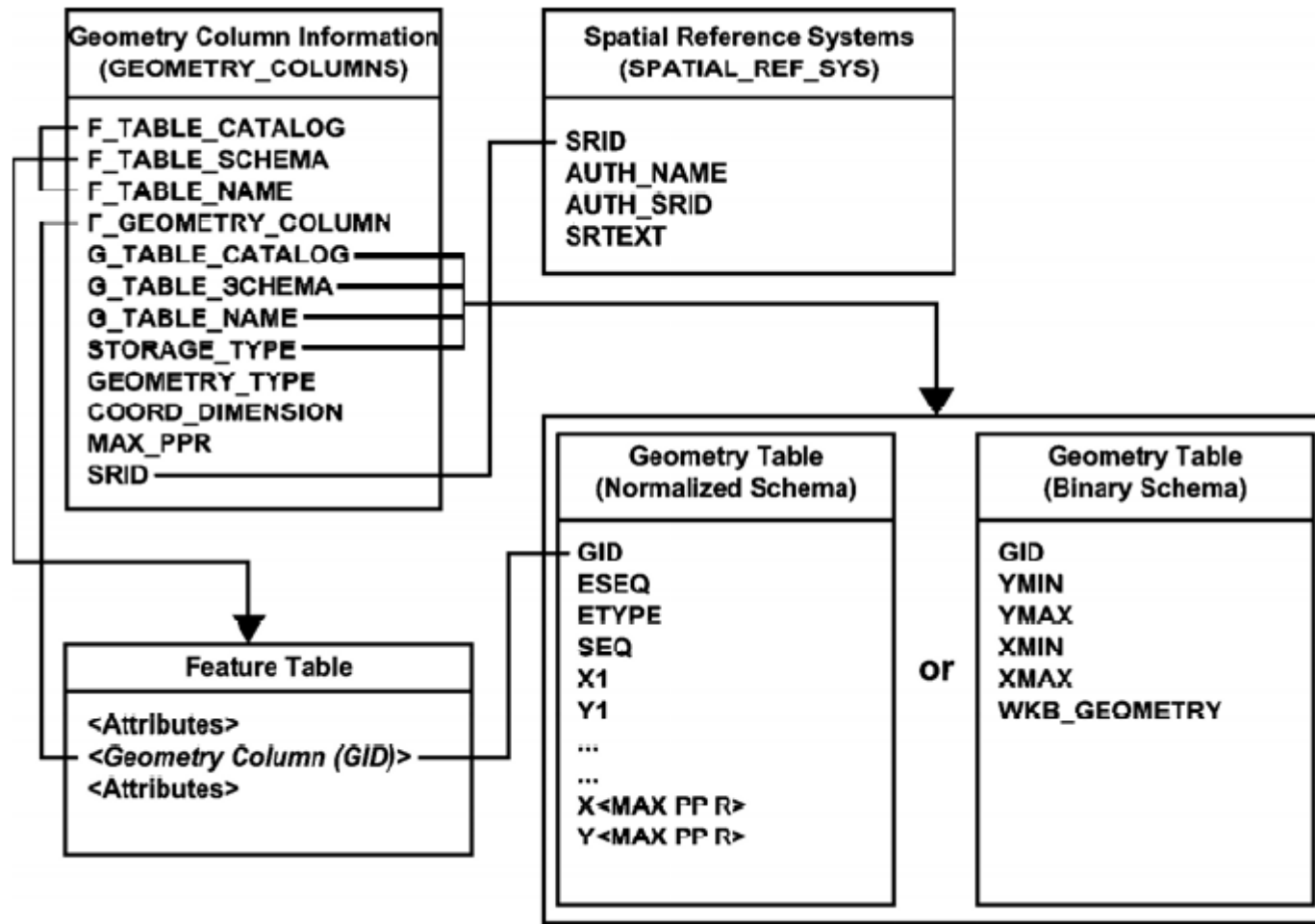
- SPATIAL_REF_SYS: Es el diccionario de códigos de sistemas de referencia espaciales (SRS). Solamente es necesaria para hacer operaciones de re-proyección, pero resulta útil para consulta durante el desarrollo. Dentro de esta tabla, interesan particularmente los campos SRID y STEXT.
 - Ej. Queremos obtener el SRID de la proyección que corresponde al datum Yacaré (

```
SELECT srid FROM spatial_ref_sys WHERE srtext LIKE  
'%Yacare%';
```


srid = 4309



SFS - SQL



SFS - SQL

Table 4: Geometry type codes

Code	Geometry type	Coordinates
0	GEOMETRY	\\ IN X Y
1	POINT	\\ IN X Y
2	LINESTRING	\\ IN X Y
3	POLYGON	\\ IN X Y
4	MULTIPOINT	\\ IN X Y
5	MULTILINESTRING	\\ IN X Y
6	MULTIPOLYGON	\\ IN X Y
7	GEOMCOLLECTION	\\ IN X Y
13	CURVE	\\ IN X Y
14	SURFACE	\\ IN X Y
15	POLYHEDRALSURFACE	\\ IN X Y
1000	GEOMETRYZ	\\ IN X Y Z
1001	POINTZ	\\ IN X Y Z
1002	LINESTRINGZ	\\ IN X Y Z

Code	Geometry type	Coordinates
1003	POLYGONZ	\\ IN X Y Z
1004	MULTIPOINTZ	\\ IN X Y Z
1005	MULTILINESTRINGZ	\\ IN X Y Z
1006	MULTIPOLYGONZ	\\ IN X Y Z
1007	GEOMCOLLECTIONZ	\\ IN X Y Z
1013	CURVEZ	\\ IN X Y M
1014	SURFACEZ	\\ IN X Y M
1015	POLYHEDRALSURFACEZ	\\ IN X Y Z
2000	GEOMETRY	\\ IN X Y M
2001	POINTM	\\ IN X Y M
2002	LINESTRINGM	\\ IN X Y M
2003	POLYGONM	\\ IN X Y M
2004	MULTIPOINTM	\\ IN X Y M
2005	MULTILINESTRINGM	\\ IN X Y M
2006	MULTIPOLYGONM	\\ IN X Y M
2007	GEOMCOLLECTIONM	\\ IN X Y M
2013	CURVEM	\\ IN X Y M
2014	SURFACEM	\\ IN X Y M
2015	POLYHEDRALSURFACEM	\\ IN X Y M
3000	GEOMETRYZM	\\ IN X Y Z M
3001	POINTZM	\\ IN X Y Z M
3002	LINESTRINGZM	\\ IN X Y Z M



Ejemplos de Operadores Espaciales en SQL

- ❑ Utilizaremos las siguientes *feature tables*:
 - Ciudades(gid, código, nombre, población, geom), capa de polígonos.
 - Calles (gid, código, nombre, geom), capa de líneas.
 - Hoteles (gid, nombre, dirección, estrellas, capacidad, geom) capa de puntos.
 - Rios(gid, nombre, geom), capa de líneas
- ❑ Operadores en el SELECT
 - Obtener nombre, código y área de la ciudad de Montevideo:

```
SELECT c.nombre, c.codigo, ST_AREA(c.geom) AS area  
FROM Ciudades c WHERE c.nombre='Montevideo';
```



Ejemplos de Operadores Espaciales en SQL

- Obtener nombre y longitud de la calle con codigo=223
SELECT r.nombre, **ST_LENGTH**(r.geom) AS longitud
FROM Calles r WHERE r.codigo=223;
- Listar nombre y densidad de población de ciudades en orden decreciente de densidad:
SELECT c.nombre, c.poblacion/**ST_AREA**(c.geom) AS densidad
FROM Ciudades c;
ORDER BY densidad DESC;



Ejemplos de Operadores Espaciales en SQL

❑ Operadores en el WHERE (Join Espacial):

- Obtener la cantidad de hoteles 4 estrellas en la ciudad de Colonia:

```
SELECT COUNT(h.nombre) FROM Hoteles h, Ciudades c
WHERE ST_CONTAINS(c.geom, h.geom)
AND c.nombre='Colonia' AND h.estrellas=4
```

- Listar todas las ciudades que se encuentren a menos de 100 km del río Uruguay

```
SELECT c.nombre FROM Ciudades c, Rios r
WHERE ST_OVERLAPS(c.geom, ST_BUFFER(r.geom, 100))
AND r.nombre='Uruguay'
```



Referencias

- ❑ Simple Features Standard

<http://www.opengeospatial.org/standards/sfs>

