Corso

di

Basi di Dati Spaziali

Estensione spaziale di SQL (i sistemi PostgreSQL e Oracle Spatial)

Angelo Montanari

Donatella Gubiani

Storia

- Storia:
  - 1974: prima proposta SEQUEL
  - 1981: prime implementazioni
  - 1983: standard di fatto
  - 1986, 1989, 1992 e 1999: standard

Funzionalità

- Insieme di linguaggi con varie funzionalità:
  - DDL
  - DML
    - aggiornamento
    - interrogazione

Per trattare dati spaziali…

- Estensione SQL con nuovi costrutti per trattare i tipi di dato spaziale introdotti
  - DDL: definizione
  - DML: aggiornamento e interrogazione

Componente DDL

- Permette di creare schemi di basi di dati: domini, tabelle, indici, asserzioni, viste, privilegi
- Meccanismi standard SQL

Domini e definizione di tabelle

- Gli attributi di una tabella possono essere associati, oltre ai tipi di dato tradizionali, ad uno dei tipi di dato spaziale
  - Point
  - Line
  - Polygon

Definizione di tabella

```
CREATE TABLE NAZIONE(
        Nome CHAR(20) PRIMARY KEY,
        NumAbitanti INTEGER NOT NULL,
        Geometria POLYGON NOT NULL
)

CREATE TABLE FIUME(
        Nome CHAR(20) PRIMARY KEY,
        Geometria LINE NOT NULL
)
```

Definizione degli indici

- Oltre ai tradizionali indici, rivestono un ruolo fondamentale gli indici spaziali

DML: Componente di interrogazione

- Permette di accedere all'informazione presente in una base dei dati (ma non consente di modificarla)

Tipi di interrogazione

- L'estensione spaziale SQL permette di rispondere, oltre a interrogazioni con criteri alfanumerici, a interrogazioni basate su criteri spaziali
  - Selezioni basate su relazioni spaziali rispetto a valori geometrici costanti o query interattive
  - Interrogazioni con criteri spaziali o join spaziali

Costrutti fondamentali

- **SELECT:** specifica la lista degli attributi che si vuole siano restituiti (anche funzioni aggregate)
- **FROM:** specifica le tabelle alle quali bisogna accedere per rispondere all'interrogazione
- **WHERE:** specifica le condizioni che devono essere soddisfatte dalle righe coinvolte nell'interrogazione per poter partecipare alla soluzione finale

Interrogazioni spaziali

- Utilizzano funzioni e relazioni su tipi di dato spaziale
  - Predicati utilizzati nella clausola WHERE (relazioni spaziali)
  - Funzioni applicate alle tuple del risultato (clausola SELECT)

Interrogazioni con criteri spaziali – 1

Esempio. Selezionare le nazioni confinanti con la (adiacenti alla) Francia

```
SELECT N1.Nome
FROM NAZIONE N1, NAZIONE N2
WHERE N2.Nome="Francia"
  and Touch(N1.Geometria,N2.Geometria)
```

Interrogazioni con criteri spaziali – 2

Esempio. Determinare i fiumi che attraversano la Germania

```
SELECT F.Nome
FROM FIUME F, NAZIONE N
WHERE N.Nome="Germania"
   and Cross(F.Geometria,N.Geometria)
```

Interrogazioni con criteri spaziali – 3

Esempio. Determinare l'area delle nazioni attraversate dal fiume Don

```
SELECT N.Nome, Area(N.Geometria)
FROM Nazione N, Fiume F
WHERE F.Nome="Don" and
      Cross(F.Geometria,N.Geometria)
```

Interrogazioni con criteri spaziali – 4

Esempio. Determinare la lunghezza totale dei fiumi che attraversano l'Italia

```
SELECT Sum(Length(F.Geometria))
FROM FIUME F, NAZIONE N
WHERE N.Nome="Italia"
   and Cross(F.Geometria,N.Geometria)
```

Interrogazioni con criteri spaziali – 5

Esempio. Determinare le nazioni contenute nel rettangolo @rectangle

```
SELECT Nome
FROM NAZIONE
WHERE In(Geometria,@rectangle)
```

DML: Componente di aggiornamento

- Permette di operare sulle tabelle: inserire, modificare e cancellare i dati di una base dei dati
- Operazioni di
  - inserimento: insert
  - eliminazione: delete
  - modifica: update

  di una o più tuple di una relazione sulla base di una condizione che può coinvolgere anche altre relazioni

PostgreSQL

- 1996 è diventato un progetto open source
- Supporta un modello *object-relational* che gestisce i tipi di dato spaziale

Estensione spaziale

- Uno dei primi sistemi a prevedere un'estenzione spaziale
  - un insieme di tipi geometrici
  - un insieme di operatori e funzioni sui tipi geometrici
  - un'estensione SQL integrata con operatori spaziali

Tipi di dato spaziale

Tipo geometrico	Rappresentazione	Descrizione
point	(x,y)	Punto
line	((x1,y1),(x2,y2))	Retta infinita
lseg	((x1,y1),(x2,y2))	Segmento
box	((x1,y1),(x2,y2))	Rettangolo
path	((x1,y1),…,(xn,yn))	Linea chiusa
path	[(x1,y1),…,(xn,yn)]	Linea (aperta)
polygon	((x1,y1),…,(xn,yn))	Poligono
circle	<(x,y),r>	Cerchio

Rappresentazione dei dati in PostgreSQL - 1

Alcune rappresentazioni sono ambigue (non identificano il corrispondente tipo di dato spaziale). Per tale ragione, il **tipo di dato** viene indicato in modo **esplicito**:

'<rappresentazione>'::tipo

Esempio. Un *rettangolo* è rappresentato da:

'((0,0),(1,1))':: box

mentre un *segmento* è rappresentato da:

'((0,0),(1,1))':: lseg

Rappresentazione dei dati in PostgreSQL - 2

La **sintassi** usata per definire un poligono è molto simile (differisce solo nel nome del tipo di dato spaziale) a quella impiegata per definire una poli-linea chiusa

Poligoni e poli-linee chiuse differiscono in:

- **funzioni** che possono essere applicate
- modalità di **memorizzazione**

Rappresentazione dei dati in PostgreSQL - 3

Il modello spaziale di PostgreSQL non contiene **né** il tipo di dato regione, inteso come insieme di poligoni (anziché singolo poligono), **né** il tipo di dato linea, inteso come insieme di linee/spezzate

Conseguenze:

- non è definita l'intersezione di poligoni, perché potrebbe generare un insieme di poligoni anziché un singolo poligono (non è chiusa)
- non è definita l'intersezione di linee e poligoni, perché potrebbe generare un insieme di linee anziché una singola linea (non è chiusa)

Rappresentazione dei dati in PostgreSQL - 4

Predicati **vs.** funzioni: gli operatori spaziali di PostegreSQL sono per la maggior parte **predicati** spaziali, anziché **funzioni**, e non permettono di costruire nuovi oggetti

Alcune operazioni sono **polimorfe**: possono essere applicate ad argomenti di diverso tipo (spaziale).
In tutte le situazioni in cui il tipo dell'operando è un qualsiasi tipo di dato spaziale, viene usato il tipo generico *geom*

Operatori spaziali - 1

Operatore	Descrizione	Sintassi
+	Trasla un rettangolo	boxXpoint→box
*	Scala un rettangolo	boxXpoint→box
#	Calcola l'intersezione di due segmenti	lsegXlseg→point
#	Restituisce il numero di vertici di un poligono	polygon→integer
&&	Determina se due poligoni sono sovrapposti	polygonXpolygon→ bool
<<	Verifica se è a destra	geomXgeom→bool
>>	Verifica se è a sinistra	geomXgeom→bool

Operatori spaziali - 2

Operatore	Descrizione	Sintassi
<^	Verifica se è sopra	geomXgeom→bool
>^	Verifica se è sotto	geomXgeom→bool
?#	Stabilisci se due linee o una linea e un poligono si intersecano	pathXgeom→bool
?-	Stabilisci se due punti definiscono un segmento orizzontale	pointXpoint→bool
?--	Stabilisci se due punti definiscono un segmento verticale	pointXpoint→bool
?-	Stabilisci se due segmenti sono perpendicolari	lsegXlseg→bool

Operatori spaziali - 3

Operatore	Descrizione	Sintassi
?||	Stabilisci se due segmenti sono paralleli	lsegXlseg→bool
@	Stabilisci se un punto è contenuto in una geometria	pointXgeom→bool
~=	Stabilisci se due poligoni sono uguali	polygonXpolygon→ bool

Funzioni spaziali

Funzione	Descrizione	Output
Area(object)	Calcola l'area di un oggetto	double precision
Box_intersect (box,box)	Determina il rettangolo di intersezione fra due rettangoli	box
Center(object)	Determina il baricentro di un oggetto	point
Diameter(circle)	Calcola il diametro di un cerchio	double precision
Height(box)	Calcola l'altezza di un rettangolo	double precision
Length(object)	Calcola la lunghezza di un oggetto	double precision

DDL: Creazione di tabelle

- Utilizza la sintassi standard SQL con la possibilità di definire attributi di tipo geometrico

DDL: esempio tabelle

```
CREATE TABLE Terreno(
        nome varchar(15);
        utilizzo varchar(10);
        geometria polygon;
        PRIMARY KEY (nome));
CREATE TABLE Strade(
        nome varchar(15);
        geometria path;
        PRIMARY KEY (nome));
```

DDL: Creazione di indici spaziali

- Creazione di indici spaziali di tipo R-Tree sugli attributi spaziali

```
CREATE INDEX RtrTerreno
        ON TERRENO USING RTREE(geometria);
```

DML: aggiornamento

- L'inserimento, la modifica e l'aggiornamento vengono effettuati utilizzando la sintassi standard SQL, tenendo conto della specifica sintassi dei diversi tipi

```
INSERT INTO
    TERRENO(nome, utilizzo, geometria)
    VALUES ('P1', 'prato', '((1,1),(2,3),(5,2))');
```

DML: interrogazione - 1

Esempio. Determinare le strade che intersecano il terreno ‘P1’

```
SELECT S.nome
FROM STRADA S, TERRENO T
WHERE T.nome=‘P1’ and
        S.geometria?#T.geometria;
```

DML: interrogazione - 2

Esempio. Determinare i terreni che stanno a destra del terreno ‘P2’

```
SELECT T1.nome
FROM TERRENO T1, TERRENO T2
WHERE T2.nome=‘P2’ and
      T2.geometria<<T1.geometria;
```

DML: interrogazione - 3

Esempio. Determinare i terreni che si sovrappongono al poligono con vertici (1,2),(9,5),(4,6),(2,5)

```
SELECT name
FROM TERRENO
WHERE '((1,2),(9,5),(4,6),(2,5))'::polygon
        && geometria;
```

DML: interrogazione - 4

Esempio. Determinare il terreno che contiene il punto (4,3) e la relativa area

```
SELECT name, area(geometria)
FROM TERRENO
WHERE '(4,3)'::point @ geometria;
```

ORACLE SPATIAL

- L'estensione *Spatial* permette di memorizzare dati spaziali
- Supporta il modello *Object-Relational* per gestire i diversi tipi di dato spaziale

Sistema di riferimento spaziale

- Oracle Spatial assume le coordinate cartesiane come default
- Possono essere utilizzati altri sistemi:
  - coordinate geodetiche
  - coordinate proiettate
  - coordinate locali
- È possibile utilizzare dei sistemi di riferimento definiti dall‘utente

I tipi di dato spaziale

- Due tipi di dato:
  - SDO_GEOMETRY
  - SDO_GEORASTER

SDO_GEOMETRY

- La geometria che descrive un oggetto spaziale è memorizzata in una colonna che assume valori di tipo SDO_GEOMETRY
- Tabelle di questo tipo sono chiamate *tabelle geometriche*

SDO_GEORASTER

- Nuovo tipo di dato introdotto in Oracle 10g
- Una griglia raster (o un'immagine) è memorizzata in una colonna che assume valori di tipo SDO_GEORASTER
- Tabelle di questo tipo sono chiamate *tabelle GeoRaster*

Concentriamoci sulla rappresentazione vettoriale e quindi sull'oggetto SDO_GEOMETRY

Modello dei dati

- La rappresentazione dei dati spaziali corrisponde a una struttura gerarchica
  - elementi: blocco base che costituisce una geometria; i tipi di elementi spaziali supportati sono i punti, le linee e i poligoni
  - geometrie: rappresentazione di un valore geometrico come un insieme ordinato di elementi
  - layer: collezione di geometrie che hanno lo stesso insieme di attributi

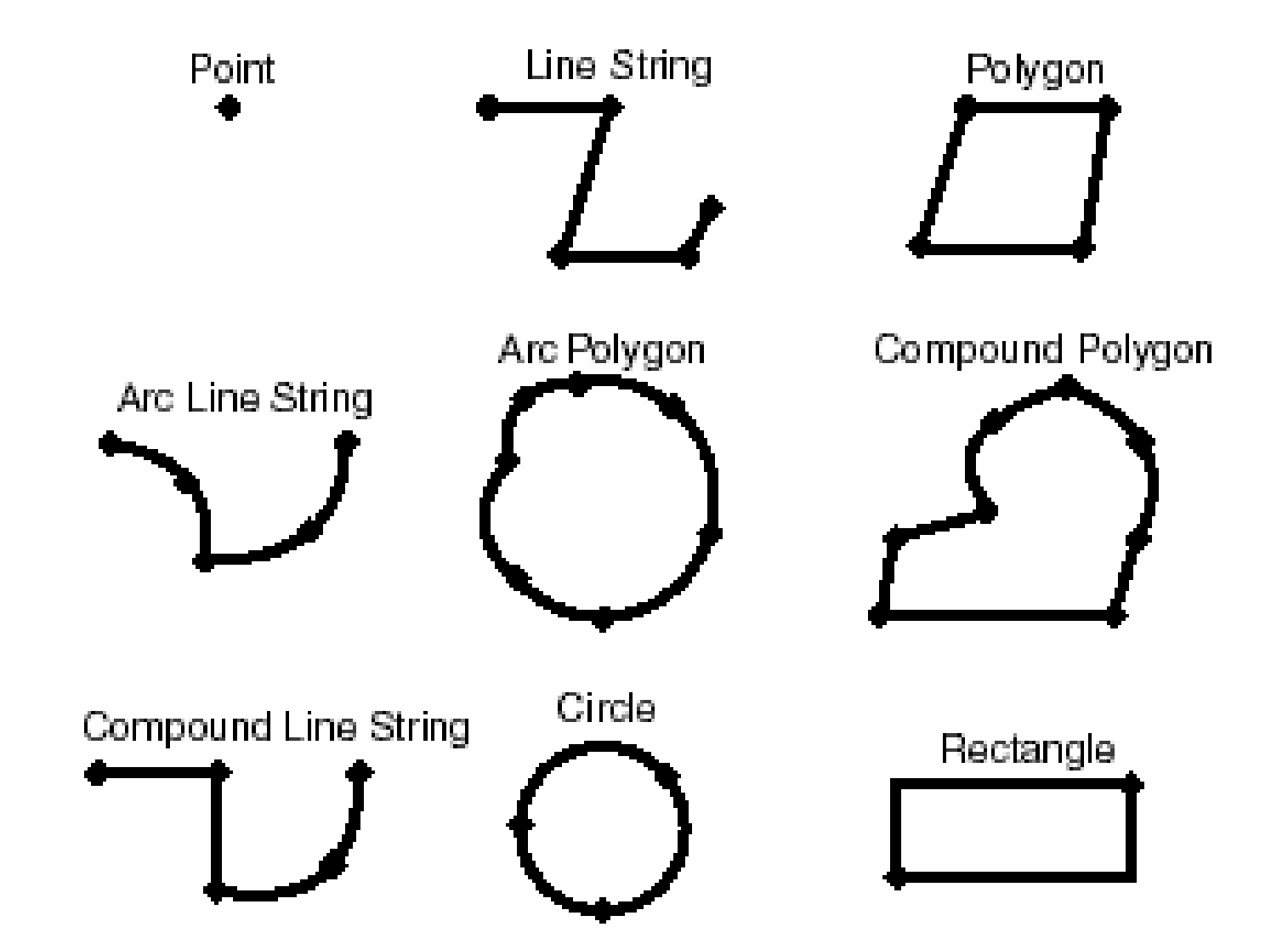
Oggetto SDO_GEOMETRY

```
CREATE TYPE SDO_GEOMETRY AS OBJECT(
   SDO_GTYPE NUMBER,
   SDO_SRID NUMBER,
   SDO_POINT SDO_POINT_TYPE,
   SDO_ELEM_INFO
       SDO_ELEM_INFO_ARRAY,
   SDO_ORDINATES
       SDO_ORDINATE_ARRAY);
```

SDO_GTYPE

- Indica il tipo di geometria
- È composto da 4 cifre nel formato *dltt*
  - *d* identifica il numero di dimensioni (2, 3, o 4)
  - *l* identifica la misura di riferimento lineare per una geometria 3-dimensionale LRS (Linear Referencing System)
  - *tt* identifica il tipo geometrico (da 00 a 07)

Esempi di valori geometrici



Valori validi per SDO_GTYPE

Valore	Tipo di geometria	Descrizione
*dl*00	UNKNOWN_GEOMETRY	Spatial ignora questa geometria
*dl*01	POINT	Un punto
*dl*02	LINE o CURVE	Stringa di linee (segmenti retti o archi circolari)
*dl*03	POLYGON	Poligoni con o senza buchi
*dl*04	COLLECTION	Collezione eterogenea di oggetti
*dl*05	MULTIPOINT	Uno o più punti
*dl*06	MULTILINE o MULTICURVE	Una o più stringhe di linee
*dl*07	MULTIPOLYGON	Poligoni multipli o disgiunti

SDO_SRID

- Può essere usato per identificare un sistema di coordinate associato alla geometria
- *null* indica che nessun sistema di coordinate è associato alla geometria (e quindi si assume il sistema di default)
- Tutte le geometrie di una colonna devono avere lo stesso SDO_SRID

SDO_POINT

- Identifica i punti, altrimenti è *null*
- Non è *null* se i vettori SDO_ELEM_INFO e SDO_ORDINATES sono entrambi *null*
- È definito usando un oggetto con attributi X, Y e Z
  - X e Y sono le coordinate di una geometria di tipo punto nello spazio bidimensionale

NOTA: il punto può essere descritto sia utilizzando SDO_POINT che i vettori SDO_ELEM_INFO e SDO_ORDINATES

SDO_ELEM_INFO

- Permette di interpretare le coordinate memorizzate in SDO_ORDINATES
- È un vettore di lunghezza variabile di triplette
  - *SDO_STARTING_OFFSET*: indica l'offset all'interno del vettore SDO_ORDINATES dove è memorizzata la prima coordinata (≥1)
  - *SDO_ETYPE*: indica il tipo dell'elemento che può essere semplice (1,2,1003,2003) o composto (4,1005,2005)
  - *SDO_INTERPRETATION*: se SDO_ETYPE indica un elemento semplice, determina l'interpretazione della sequenza, altrimenti specifica quante sono le triplette successive

Valori validi per SDO_ELEM_INFO - 1

SDO_ETYPE	SDO_INTERP.	Significato
0	qualsiasi valore numerico	Modella tipi geometrici non supportati da Oracle Spatial
1	1	Tipo punto
1	n>1	Gruppo di n punti
2	1	Stringa di linee i cui vertici sono connessi da segmenti di linee rette
2	2	Stringa di linee formata da una sequenza di archi circolari

Valori validi per SDO_ELEM_INFO - 2

SDO_ETYPE	SDO_INTERP.	Significato
1003 o 2003	1	Poligono semplice i cui vertici sono connessi da segmenti di linee rette
1003 o 2003	2	Poligono formato da una sequenza connessa di archi circolari
1003 o 2003	3	Tipo rettangolo: solo due punti (basso a sinistra e alto a destra)
1003 o 2003	4	Tipo cerchio: tre punti tutti appartenenti alla circonferenza

Valori validi per SDO_ELEM_INFO - 3

SDO_ETYPE	SDO_INTERP.	Significato
4	n>1	Stringa di linee composte: n indica il numero di triplette successive, le successive n triplette descrivono ogni sottoelemento
1005 o 2005	n>1	Poligono composto: n indica il numero di triplette successive, le successive n triplette descrivono ogni sottoelemento

NOTA: in 1003/1005 e 2003/2005,il valore 1 iniziale identifica il bordo esterno e il valore 2 identifica quello interno (nel caso di poligoni con buchi)

SDO_ORDINATES

- È un vettore di lunghezza variabile di oggetti di tipo NUMBER
- Memorizza le coordinate che definiscono la frontiera di un valore geometrico

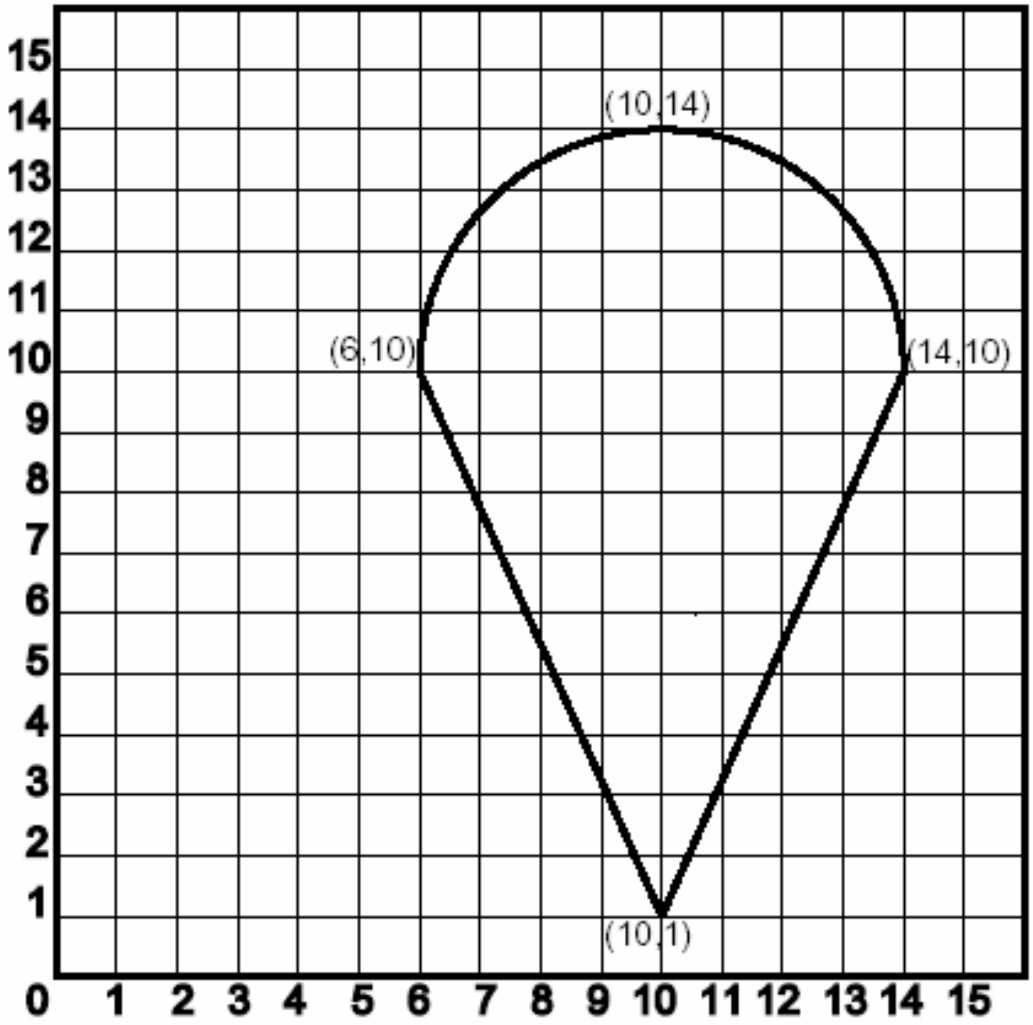
SDO_GEOMETRY: Esempio - 1

- Creare una tabella geografica EDIFICI in cui memorizzare il numero identificativo, il nome e la forma degli edifici di una città

```
CREATE TABLE edifici(
    id NUMBER PRIMARY KEY,
    nome VARCHAR2(32),
    estensione SDO_GEOMETRY
);
```

SDO_GEOMETRY: Esempio - 2

- Inserire la seguente istanza



SDO_GEOMETRY: Esempio - 3

- SDO_GTYPE = 2003
  - d=2 indica una geometria bidimensionale
  - tt=03 indicano che si tratta di un poligono
- SDO_SRID = null
  - nessun sistema di coordinate è stato esplicitamente assegnato alla geometria
- SDO_POINT = null
  - ignorato da Spatial perché non si tratta di una geometria di tipo punto

SDO_GEOMETRY: Esempio - 4

- SDO_ELEM_INFO = (1,1005,2, 1,2,1, 5,2,2)
  - prima tripletta: indica che si tratta di un poligono composto da due segmenti di linee descritti nelle triplette successive
  - seconda tripletta: indica che la prima parte del poligono è costituito da segmenti di linee rette e le relative coordinate iniziano all'offset 1
  - terza tripletta: indica che la seconda stringa di linee è composta da un arco circolare le cui coordinate si trovano a partire dall'offset 5

SDO_GEOMETRY: Esempio - 5

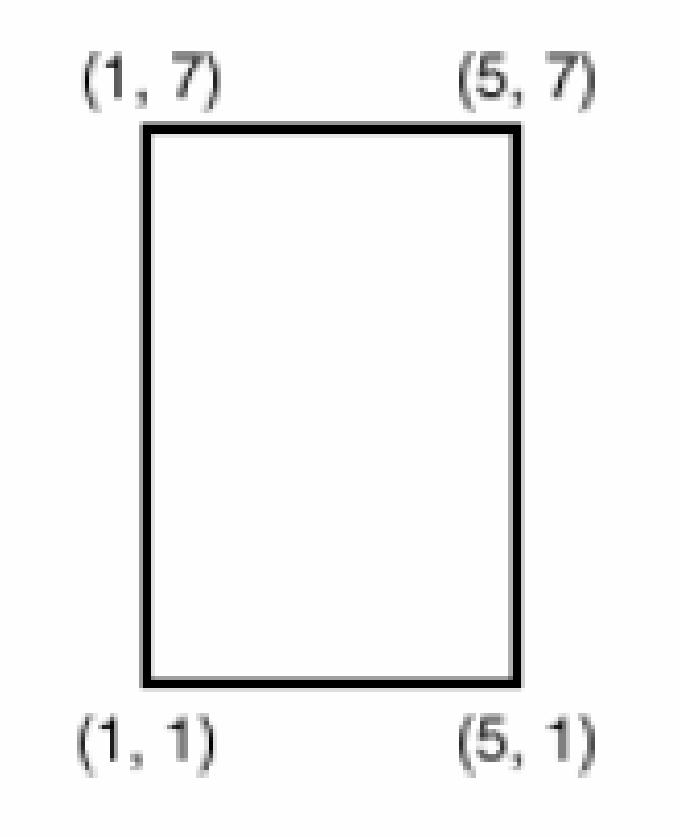
- SDO_ORDINATES =
  (6,10, 10,1, 14,10, 10,14, 6,10)
  - indica le coordinate dei punti del bordo del poligono che devono essere interpretate come descritto in SDO_ELEM_INFO

SDO_GEOMETRY: Esempio - 6

```
INSERT INTO edifici VALUES(
   12,
   'Palazzo del comune',
   SDO_GEOMETRY(
      2003, null, null,
      SDO_ELEM_INFO_ARRAY(1,1005,2, 1,2,1, 5,2,2),
      SDO_ORDINATE_ARRAY(6,10, 10,1, 14,10,
                                    10,14, 6,10)
      )
    );
```

SDO_GEOMETRY: Esempio - 7

- Inserire la seguente istanza



SDO_GEOMETRY: Esempio - 8

- SDO_GTYPE = 2003
  - d=2 indica una geometria bidimensionale
  - tt=03 indicano che si tratta di un poligono
- SDO_SRID = null
  - nessun sistema di coordinate è stato esplicitamente assegnato alla geometria
- SDO_POINT = null
  - ignorato da Spatial perché non si tratta di una geometria di tipo punto

SDO_GEOMETRY: Esempio - 9

- SDO_ELEM_INFO = (1,1003,3)
  - 1003,3: indica che si tratta di un poligono di tipo rettangolo
- SDO_ORDINATES = (1, 1, 5, 7)
  - indentifica le coordinate del vertice in basso a sinistra e di quello in alto a destra

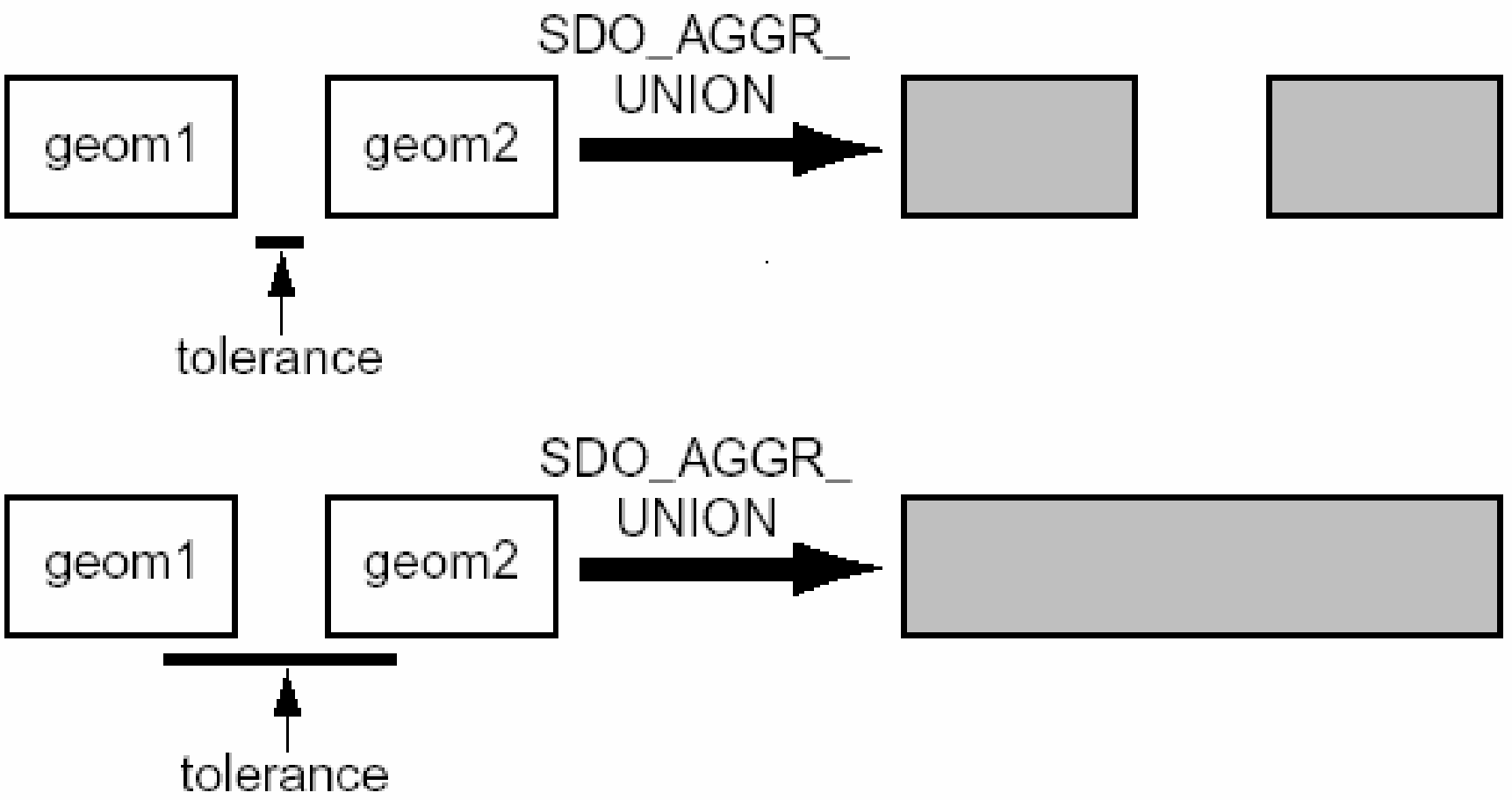
SDO_GEOMETRY: Esempio - 10

```
INSERT INTO edifici VALUES(
   8,
   'Chiesa',
   SDO_GEOMETRY(
      2003, null, null,
      SDO_ELEM_INFO_ARRAY (1,1003,3),
      SDO_ORDINATE_ARRAY (1, 1, 5, 7)
      )
   );
```

Tolleranza - 1

- Il valore *tolerance* è utilizzato per associare un livello di precisione ai dati spaziali
- È specificata in due casi:
  - nella definizione dei metadati
  - nella definizione di alcune funzioni

Tolleranza - 2



Funzioni geometriche

- Funzioni associate al tipo di dato SDO_GEOMETRY
- Diverse categorie:
  - funzioni di verifica delle relazioni che intercorrono tra due oggetti
  - funzioni di validazione
  - operazioni su singoli oggetti
  - operazioni su coppie di oggetti

Funzioni geometriche: relazioni fra due oggetti

- SDO_GEOM.RELATE Stabilisce se due oggetti stanno in una determinata relazione (topologica)
- SDO_GEOM.WITHIN_DISTANCE Determina se due oggetti geometrici si trovano ad una distanza inferiore ad un valore prefissato

Funzioni geometriche: funzioni di validazione

- SDO_GEOM. VALIDATE_GEOMETRY_WITH_CONTEXT
  Verifica se una geometria è valida
- SDO_GEOM. VALIDATE_LAYER_WITH_CONTEXT
  Verifica se tutte le geometrie memorizzate in una colonna sono valide

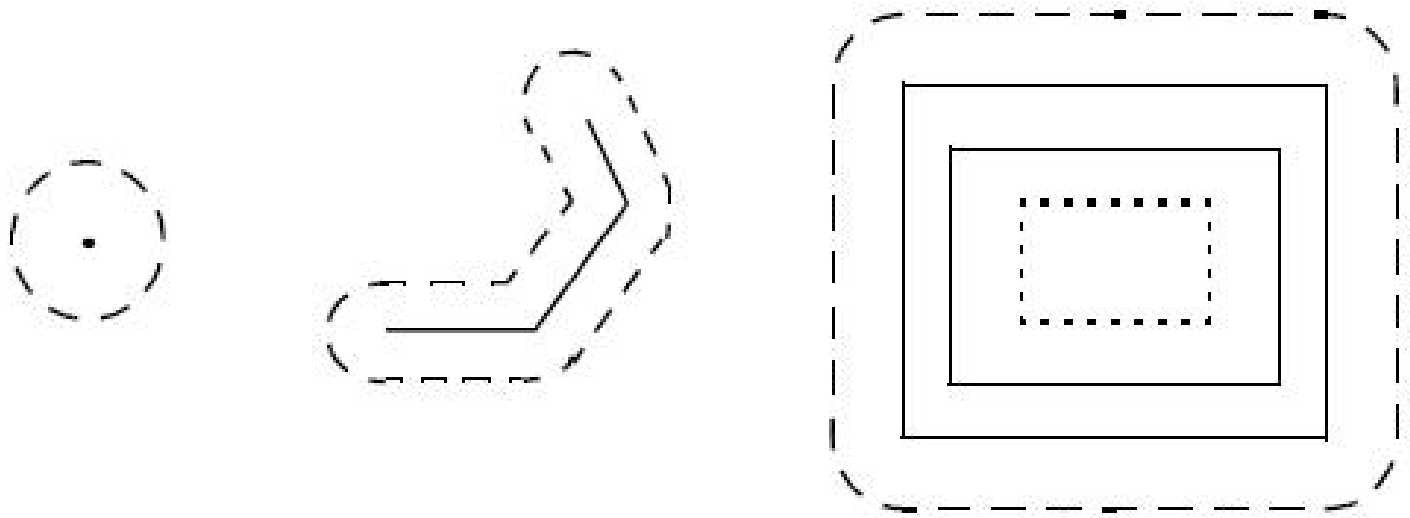
NOTA: in alcuni casi i controlli sono effettuati in modo automatico (caso dei rettangoli)

Funzioni geometriche: operazioni su singoli oggetti -1

- SDO_GEOM.SDO_ARC_DENSIFY Approssima ogni arco circolare con una spezzata aperta ed ogni cerchio con una spezzata chiusa (la qualità dell'approssimazione è specificata attraverso opportuni parametri)
- SDO_GEOM.SDO_AREA Calcola l'area di un poligono bidimensionale

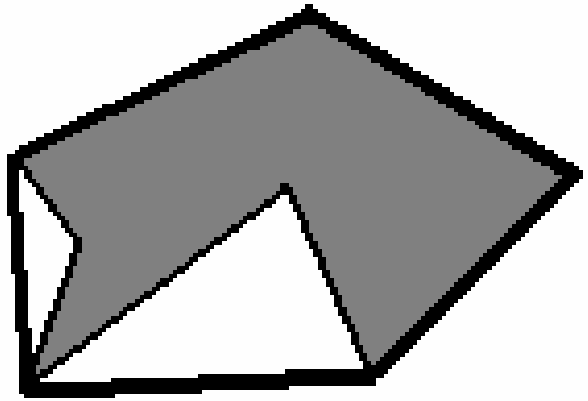
Funzioni geometriche: operazioni su singoli oggetti -2

- SDO_GEOM.SDO_BUFFER Genera un poligono di buffer attorno ad una geometria



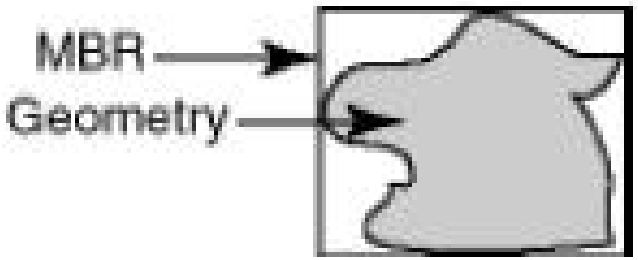
Funzioni geometriche: operazioni su singoli oggetti -3

- SDO_GEOM.SDO_CONVEXHULL
  Ritorna un oggetto di tipo poligono che rappresenta il bordo convesso della geometria



Funzioni geometriche: operazioni su singoli oggetti -4

- SDO_GEOM.SDO_MBR Ritorna il minimum bounding rectangle (MBR) di una geometria



Funzioni geometriche: operazioni su singoli oggetti -5

- SDO_GEOM.SDO_MAX_MBR_ORDINATE
  Ritorna il massimo valore per l'ordinata specificata del minimum bounding rectangle di una geometria
- SDO_GEOM.SDO_MIN_MBR_ORDINATE
  Ritorna il minimo valore per l'ordinata specificata del minimum bounding rectangle di una geometria

Funzioni geometriche: operazioni su singoli oggetti -6

- SDO_GEOM.SDO_LENGTH Calcola la lunghezza, o perimetro, della geometria
- SDO_GEOM.SDO_CENTROID Ritorna il centroide di un poligono
- SDO_GEOM.SDO_POINTSURFACE Ritorna un punto che sicuramente appartiene alla superficie di un poligono

Funzioni geometriche: operazioni su due oggetti - 1

- SDO_GEOM.SDO_DISTANCE Calcola la distanza fra due geometrie
- SDO_GEOM.SDO_DIFFERENCE Ritorna una geometria che è la differenza topologica (operazione MINUS) di due oggetti geometrici
- SDO_GEOM.SDO_INTERSECTION Ritorna una geometria che è l'intersezione topologica (operazione AND) di due oggetti geometrici

Funzioni geometriche: operazioni su due oggetti - 2

- SDO_GEOM.SDO_UNION Ritorna una geometria che è l'unione topologica (operazione OR) di due oggetti geometrici
- SDO_GEOM.SDO_XOR Ritorna una geometria che è la differenza simmetrica topologica (operazione XOR) di due oggetti geometrici

Funzioni geometriche

Esempio. Data la tabella TERRITORIO (id, estensione) calcolare l'area di ogni istanza

```
SELECT id,
SDO_GEOM.SDO_AREA(estensione,0.005)
FROM TERRITORIO
```

Relazioni topologiche - 1

- Oracle Spatial definisce le relazioni topologiche utilizzando il modello 9-intersection
- Non vengono esplicitamente memorizzate nelle tabelle, ma sono derivate per mezzo dell'operatore SDO_RELATE

Relazioni topologiche - 2

- DISJOINT I contorni e gli interni non si intersecano
- TOUCH I contorni si intersecano, gli interni no
- OVERLAPBDYDISJOINT L'interno di un oggetto interseca il contorno e l'interno dell'altro oggetto, i due contorni non si intersecano
- OVERLAPBDYINTERSECT I contorni e gli interni dei due oggetti si intersecano

Relazioni topologiche - 3

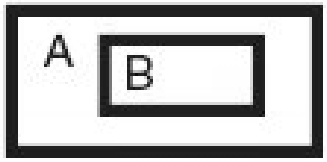
- EQUAL I due oggetti hanno lo stesso contorno ed interno
- CONTAINS L'interno ed il contorno di un oggetto sono completamente contenuti nell'interno dell'altro oggetto
- COVERS L'interno, ma non il contorno, di un oggetto è completamente contenuto nell'interno dell'altro oggetto
- INSIDE Inverso di CONTAINS

Relazioni topologiche - 4

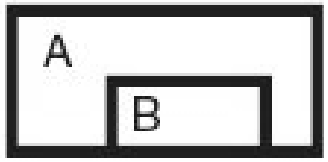
- COVERED BY Inverso di COVERS
- ON L'interno ed il contorno di un oggetto appartengono al contorno dell'altro oggetto
- ANYINTERACT Gli oggetti sono non disgiunti

NOTA: le relazioni touch, equal, overlapbdydisjoint, overlapbdyintersect, disjoint sono simmetriche

Relazioni topologiche - 5

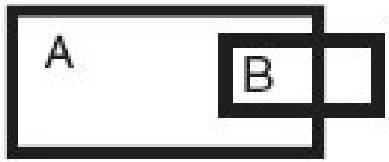


A CONTAINS B
B INSIDE A

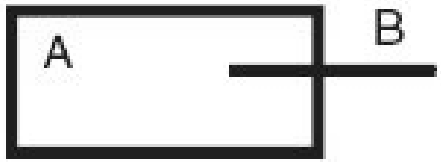


A COVERS B
B COVEREDBY A

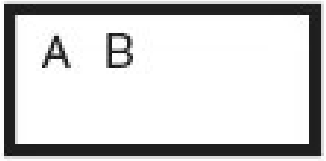
A TOUCH B
B TOUCH A



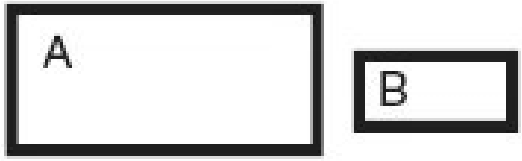
A OVERLAPBDYINTERSECT B
B OVERLAPBDYINTERSECT A



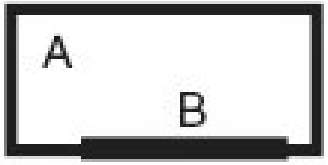
A OVERLAPBDYDISJOINT B
B OVERLAPBDYDISJOINT A



A EQUAL B
B EQUAL A



A DISJOINT B
B DISJOINT A



B ON A

Interrogazioni spaziali

- Per la risoluzione di interrogazioni spaziali Oracle Spatial sfrutta il modello "two-tier query"
- Sono utilizzate due distinte operazioni:
  - *filtro primario*: permette di selezionare i candidati che vanno passati al secondo filtro ed utilizza la geometria (o indice) per ridurre la complessità computazionale
  - *filtro secondario*: applicato al resultato del filtro primario, restituisce il risultato esatto dell'interrogazione

Operatori spaziali - 1

- SDO_FILTER Stabilisce se gli MBR di due oggetti geometrici stanno in una data relazione topologica (lavoro svolto dal primo filtro)
- SDO_NN Determina l'oggetto/gli oggetti geometrici più vicini ad un dato oggetto geometrico
- SDO_NN_DISTANCE Restituisce la distanza di un oggetto identificato con SDO_NN

Operatori spaziali - 2

- SDO_RELATE Determina se due oggetti geometrici stanno in una data relazione topologica
- SDO_WITHIN_DISTANCE Determina se due geometria sono entro una specifica distanza l'uno dall'altro
- SDO_JOIN Esegue un join spaziale basato su una o più relazioni topologiche

SDO_RELATE - 1

- Determina se un oggetto geometrico è in una particolare relazione topologica con un altro oggetto geometrico, quale, ad esempio, un'area di interesse
- Sintassi:

SDO_RELATE(geometria1,geometria2,parametri)

SDO_RELATE - 2

- La geometria1 specifica una colonna geometrica di una tabella
- La geometria2 può specificare sia una colonna geometrica che un valore geometrico

SDO_RELATE - 3

- Parametri:
  - **mask**: specifica la relazione topologica di interesse ed è obbligatorio
    - è possibile combinare i valori
      es. 'mask=inside+touch'
  - **min_resolution** e/o **max_resolution**: è opzionale e specifica che devono essere considerate solo le geometrie il cui MBR ha almeno un lato di lunghezza uguale o maggiore e/o minore o uguale al valore specificato

SDO_RELATE - 4

- Deve essere usato all‘interno della clausola WHERE
- Ritorna vero se il confronto tra le coppie di oggetti geometrici sono nella relazione topologica specificata da mask, falso altrimenti

SDO_RELATE - 5

Esempio. Data la tabella geometrica così definita EDIFICIO (id, descrizione, estensione) determinare la descrizione degli edifici che si trovano all'interno della geometria così definita SDO_GEOMETRY

(2003,null,null,SDO_ELEM_INFO_ARRAY(1,1003,3),

SDO_ORDINATE_ARRAY(1,1,20,20))

SDO_RELATE - 6

```
SELECT descrizione
FROM EDIFICIO
WHERE SDO_RELATE(
    estensione,
    SDO_GEOMETRY(2003, null,
    null,SDO_ELEM_INFO_ARRAY(1,
    1003,3),SDO_ORDINATE_ARRAY(
    1,1,20,20)),
    'mask=inside')='TRUE';
```

SDO_RELATE - 7

- Sono stati introdotti una serie di operatatori SDO_RELATE specifici per le diverse relazioni topologiche:
  - SDO_ANYINTERACT - SDO_CONTAINS
  - SDO_OVERLAPS - SDO_TOUCH
  - SDO_COVERS - SDO_EQUAL
  - SDO_INSIDE
  - SDO_COVEREDBYSDO_ON
  - SDO_OVERLAPBDYDISJOINT
  - SDO_OVERLAPBDYINTERSECT

SDO_WITHIN_DISTANCE - 1

- Determina se due oggetti spaziali si trovano a meno di una determinata distanza (euclidea) tra di loro
- Sintassi:
  SDO_WITHIN_DISTANCE(geometria1,
  ageom, parametri)

SDO_WITHIN_DISTANCE - 2

- La geometria1 specifica una colonna geometrica che contiene l'insieme di geometrie che sono valutate per determinare se stanno entro la distanza specificata dall'oggetto aGeom
- aGeom specifica una colonna geometrica o una variabile geometrica a partire dalla quale misurare la distanza da geometria1

SDO_WITHIN_DISTANCE - 3

- Parametri:
  - **distance**: specifica il valore della distanza ed è obbligatorio
  - **min_resolution** e/o **max_resolution**: è opzionale e specifica che devono essere considerate solo le geometrie il cui MBR ha almeno un lato di lunghezza uguale o maggiore e/o minore o uguale al valore specificato
  - **querytype**: settato a filter esegue esclusivamente il filtro primario, altrimenti (se non specificato), esegue anche il filtro secondario
  - **unit**: specifica un'unità di misura associata alla distanza

SDO_WITHIN_DISTANCE - 4

- Deve essere usato all‘interno della clausola WHERE
- Ritorna vero se la distanza tra i due oggetti geometrici specificati è minore del valore specificato dal parametro distance, falso altrimenti

SDO_WITHIN_DISTANCE - 5

Esempio. Data la tabella geometrica così definita EDIFICIO (id, descrizione, estensione) determinare la descrizione degli edifici che si trovano ad una distanza inferiore a 20m dal punto così definito SDO_GEOMETRY (2001, null, null, SDO_ELEM_INFO_ARRAY(1,1,1), SDO_ORDINATE_ARRAY(4,4))

SDO_WITHIN_DISTANCE - 6

```
SELECT descrizione
FROM EDIFICIO
WHERE SDO_WITHIN_DISTANCE (
    estensione,
    SDO_GEOMETRY(
        2001,null,null,
        SDO_ELEM_INFO_ARRAY(1,1,1),
        SDO_ORDINATE_ARRAY(4,4)),
    'distance=20')= 'TRUE';
```

SDO_JOIN - 1

- Esegue un join spaziale basato su una o più relazioni topologiche
  - non è un vero e proprio operatore spaziale ma una funzione su tabelle
- Sintassi:

SDO_JOIN(tabella1, colonna1, tabella2, colonna2, parametri, ordine)

SDO_JOIN - 2

- tabella1 e geometria1 specificano una tabella ed una sua colonna geometrica
- tabella2 e geometria2 specificano una seconda tabella ed una sua colonna geometrica

SDO_JOIN - 3

- Parametri (opzionali):
  - **mask**: specifica la relazione topologica di interesse
    - default = filter
  - **distance**: specifica una distanza di tolleranza che deve essere considerata prima di verificare il controllo topologico
  - **unit**: specifica un'unità di misura associata alla distanza
- Ordine (opzionale): specifica se l'ordine di elaborazione delle tabelle coinvolte nel join deve essere preservato

SDO_JOIN - 4

- Deve essere usato all‘interno della clausola FROM
- Ritorna un oggetto SDO_ROWIDSET, che consiste in una tabella di oggetti SDO_ROWIDPAIR

SDO_JOIN - 5

Esempio. Data la tabella geometrica EDIFICIO (id, descrizione, estensione), selezionare, per ogni edificio, tutti gli edifici con esso in relazione ANYINTERACT

SDO_JOIN - 6

```
SELECT a.descrizione, b.descrizione
FROM TABLE(SDO_JOIN('EDIFICIO',
   ,'estensione', 'EDIFICIO', 'estensione',
   'mask=ANYINTERACT')) c,
   EDIFICIO a,
   EDIFICIO b
WHERE c.rowid1 = a.rowid
        AND c.rowid2 = b.rowid
ORDER BY a.descrizione;
```

Funzioni aggregate - 1

- Le funzioni aggregate vengono utilizzate per aggregare i risultati ottenuti da un‘interrogazione
- Analoghe alle funzioni aggregate tradizionali, lavorano su dati spaziali

Funzioni aggregate - 2

- SDO_AGGREGATE_CENTROID Ritorna una geometria che è il centro di gravità (centroide) della geometria specificata
- SDO_AGGR_CONVEXHULL Ritorna una geometria che è il convex hull (bordo convesso) delle geometrie specificate
- SDO_AGGR_LRS_CONCAT Ritorna una geometria LRS (Linear Referencing System) che concatena gli oggetti geometrici LRS specificati

Funzioni aggregate - 3

- SDO_AGGR_MBR Ritorna il contorno del rettangolo minimo che contiene tutte le geometrie specificate
- SDO_AGGR_UNION Ritorna un oggetto geometrico che è l'unione topologica (operazione OR) degli oggetti geometrici specificati

Funzioni aggregate - 4

- Molte funzioni accettano in input un parametro di tipo SDOAGGRTYPE così definito:

```
CREATE TYPE SDOAGGRTYPE AS OBJECT(
        geometry SDO_GEOMETRY,
        tolerance NUMBER);
```

Funzioni aggregate - 5

- Data la tabella geometrica così definita TERRENI (id, estensione, utilizzo) determinare l'unione dei terreni utilizzati per scopi agricoli.

```
SELECT SDO_AGGR_UNION(
    SDOAGGRTYPE(t.estensione,0.005))
FROM TERRENI as t
WHERE t.utilizzo='agricolo';
```

Indici - 1

- Permettono di ottenere una maggiore efficienza nell'accesso dei dati spaziali
- Possono essere di due tipi:
  - R-Tree
  - Quadtree

Indici - 2

- Creare un indice R-Tree sull'attributo geometrico estensione della tabella territorio

  CREATE INDEX territorio_ind
  ON territorio(estensione)
  INDEXTYPE IS MDSYS.SPATIAL_INDEX;

- Per creare un indice di tipo Quadtree bisogna aggiungere un parametro

  PARAMETER ('SDO_LEVEL=8')

Indici - 3

- Se si vuole fare in modo che un attributo di tipo SDO_GEOMETRY contenga solo un tipo di geometria bisogna usare la keyword *layer_gtype* nella clausola PARAMETERS del comando CREATE INDEX

Indici - 4

- Creare un indice R-Tree sull'attributo geometrico estensione della tabella_con_poligono imponendo che la geometria sia di tipo poligono:

```
CREATE INDEX polygono_ind
ON tabella_con_poligono(estensione)
INDEXTYPE IS MDSYS.SPATIAL_INDEX
PARAMETERS ('layer_gtype=POLYGON');
```

Indici - 5

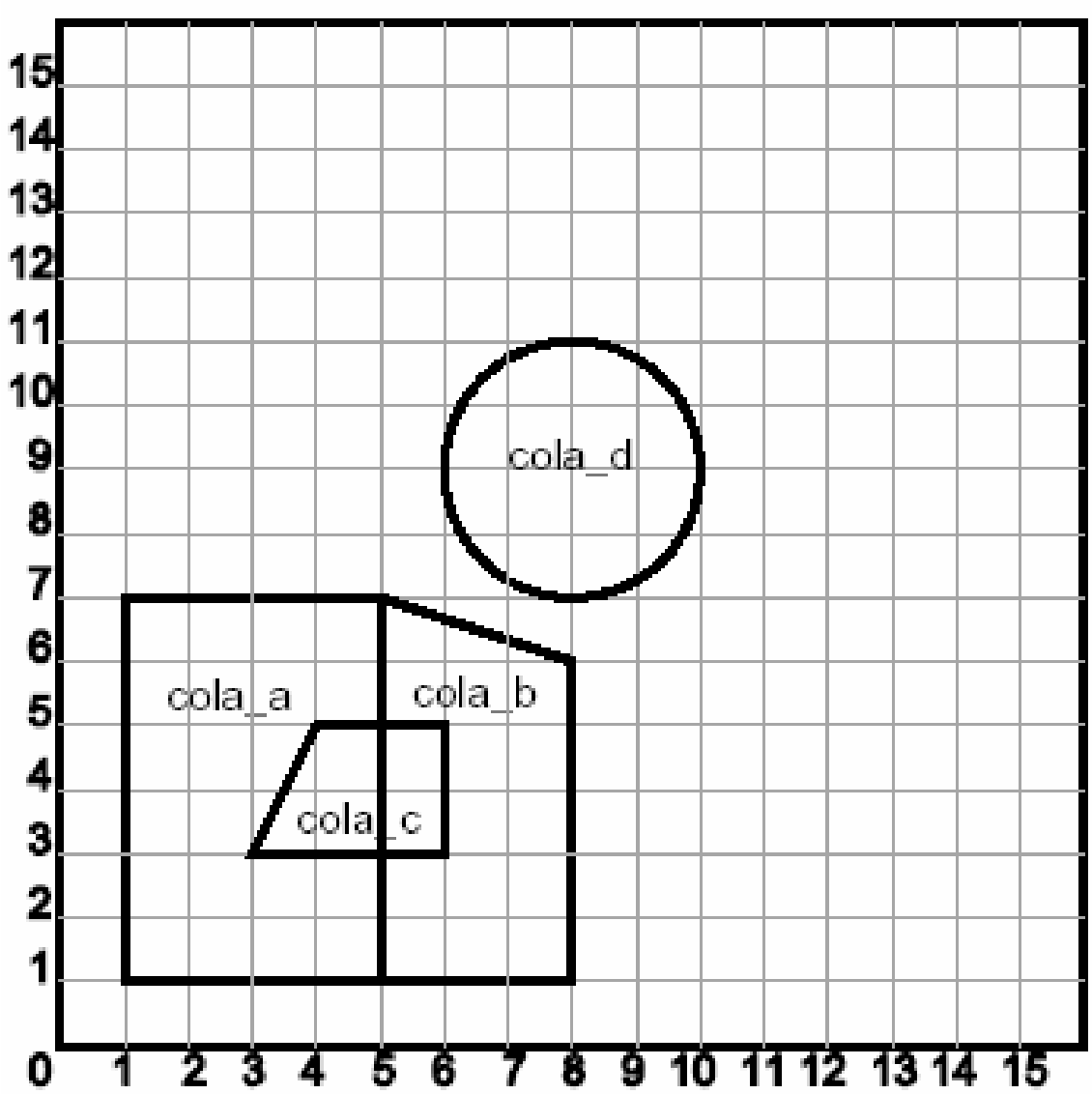
- Prima di creare un indice è necessario aggiornare la vista USER_SDO_GEOM_METADATA che descrive dimensione, estremi e tolleranza sulle diverse dimensioni

```
INSERT INTO USER_SDO_GEOM_METADATA
    VALUES (
        'nomeTabella',
        'nomeAttributoGeometrico',
        SDO_DIM_ARRAY(   -- griglia 20x20
                SDO_DIM_ELEMENT('X',0,20,0.005),
                SDO_DIM_ELEMENT('Y',0,20,0.005)),
        null);
```

Esercizio - 1

- Lo scenario è quello di un produttore di bibita analcolica che ha identificato aree geografiche di interesse dal punto di vista del marketing di diversi prodotti (cola).

Esercizio - 2


15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
cola_d
cola_a
cola_b
cola_c

Esercizio - 3

- Creazione di una tabella (COLA_MARKETS) che permette di memorizzare dati spaziali
- Aggiornamento della vista USER_SDO_GEOM_METADATA per riflettere la dimensione delle aree
- Creazione di un indice spaziale (COLA_SPATIAL_IDX)
- Inserimento delle istanze relative ai quattro oggetti spaziali (*cola_a*, *cola_b*, *cola_c*, *cola_d*)
- Esecuzione di alcune interrogazioni spaziali

Creazione

- Creazione di una tabella geografica dal nome cola_markets con tre attributi: l'identificatore (mkt_id), il nome (name) e la geometria associata (shape).

```
CREATE TABLE cola_markets (
    mkt_id NUMBER PRIMARY KEY,
    name VARCHAR2(10),
    shape SDO_GEOMETRY);
```

Indice - 1

- Aggiornamento della vista USER_SDO_GEOM_METADATA per la creazione degli indici

```
INSERT INTO USER_SDO_GEOM_METADATA
   VALUES (
        'cola_markets',
        'shape',
        SDO_DIM_ARRAY( -- 20X20 grid
                SDO_DIM_ELEMENT('X', 0, 20, 0.005),
                SDO_DIM_ELEMENT('Y', 0, 20, 0.005)),
        NULL – SRID
   );
```

Indice - 2

- Creazione dell'indice

```
CREATE INDEX cola_spatial_idx
   ON cola_markets(shape)
   INDEXTYPE IS MDSYS.SPATIAL_INDEX;
   -- Preceding created an R-tree index.
   -- Following line was for an earlier quadtree
   -- index: PARAMETERS('SDO_LEVEL = 8');
```

Inserimento

- Inserimento delle quattro istanze relative agli oggetti presenti sul piano di interesse (cola_a, cola_b, cola_c, cola_d)

```
INSERT INTO cola_markets VALUES(
    1,
    'cola_a',
    SDO_GEOMETRY(
        2003, -- poligono bidimensionale
        NULL,
        NULL,
        SDO_ELEM_INFO_ARRAY(1,1003,3),
        -- rettangolo (1003 = exterior)
        SDO_ORDINATE_ARRAY(1,1, 5,7)
        -- sono necessari solo due punti per definire
        -- un rettangolo
    ;
```

```
INSERT INTO cola_markets VALUES(
    2,
    'cola_b',
    SDO_GEOMETRY(
        2003, -- poligono bidimensionale
        NULL,
        NULL,
        SDO_ELEM_INFO_ARRAY(1,1003,1),
        -- poligono (1003=exterior)
        SDO_ORDINATE_ARRAY (5,1, 8,1, 8,6, 5,7, 5,1)
    )
);
```

```
INSERT INTO cola_markets VALUES(
    3,
    'cola_c',
    SDO_GEOMETRY(
        2003, -- poligono bidimensionale
        NULL,
        NULL,
        SDO_ELEM_INFO_ARRAY(1,1003,1),
        -- poligono (exterior)
        SDO_ORDINATE_ARRAY(3,3, 6,3, 6,5, 4,5, 3,3)
    )
);
```

```
INSERT INTO cola_markets VALUES(
    4,
    'cola_d',
    SDO_GEOMETRY(
        2003, -- poligono bidimensionale
        NULL,
        NULL,
        SDO_ELEM_INFO_ARRAY(1,1003,4),
        -- cerchio
        SDO_ORDINATE_ARRAY(8,7, 10,9, 8,11)
    )
);
```

Interrogazione - 1

- Determinare l'intersezione topologica delle geometrie di cola_a e cola_c

```
SELECT
   SDO_GEOM.SDO_INTERSECTION(
   c_a.shape, c_c.shape, 0.005)
FROM cola_markets c_a, cola_markets c_c
WHERE c_a.name='cola_a' AND
   c_c.name='cola_c';
```

Interrogazione - 2

- Determinare se le geometrie di cola_b e cola_d sono non disgiunte (anyinteract)

```
SELECT SDO_GEOM.RELATE(c_b.shape,
   'ANYINTERACT', c_d.shape, 0.005)
FROM cola_markets c_b, cola_markets c_d
WHERE c_b.name='cola_b' AND
   c_d.name='cola_d';
```

Interrogazione - 3

- Determinare l'area di tutti i prodotti cola

```
SELECT name,
  SDO_GEOM.SDO_AREA(shape,0.005)
FROM cola_markets;
```

Interrogazione - 4

- Determinare l'area di cola_a

```
SELECT c.name,
   SDO_GEOM.SDO_AREA(c.shape,0.005)
FROM cola_markets c
WHERE c.name = 'cola_a';
```

Interrogazione - 5

- Determinare la distanza fra le geometrie cola_b e cola_d

```
SELECT SDO_GEOM.SDO_DISTANCE(
   c_b.shape, c_d.shape, 0.005)
FROM cola_markets c_b, cola_markets c_d
WHERE c_b.name='cola_b' AND
   c_d.name='cola_d';
```

Interrogazione - 6

- Determinare le aree di mercato che intersecano cola_c

```
SELECT x.name
FROM cola_markets c, cola_markets x
WHERE c.name='cola_c' AND
   SDO_RELATE( x.shape, c.shape,
        'mask=OVERLAPBDYINTERSECT')=
   ‘TRUE‘;
```

Interrogazione - 7

- Determinare le geometrie che si trovano ad una distanza inferiore a 2 da cola_d

```
SELECT x.name
FROM cola_markets d, cola_markets x
WHERE d.name='cola_d' AND
   x.name<>'cola_d' AND
   SDO_WITHIN_DISTANCE(x.shape, d.shape,
   'distance=2')='TRUE';
```