Przetwarzanie danych przestrzennych

Metadane, indeksowanie, przetwarzanie

Krzysztof Jankiewicz, Marek Wojciechowski

Plan

- Metadane
- Indeksowanie
- Przetwarzanie danych przestrzennych

Metadane

- Poszczególne kolumny typu **`SDO_GEOMETRY`** mogą zostać opisane (zarejestrowane) jako warstwy dzięki umieszczeniu w bazie danych odpowiednich informacji w tzw. metadanych
- Metadane pozwalają aplikacjom podejmować decyzje odnośnie technik obsługi (np. indeksacji, wizualizacji) określonych kolumn
- Dostęp do metadanych jest możliwy za pomocą perspektyw systemowych **`[USER|ALL|DBA]_SDO_GEOM_METADATA`**
- Struktura tych perspektyw jest następująca:

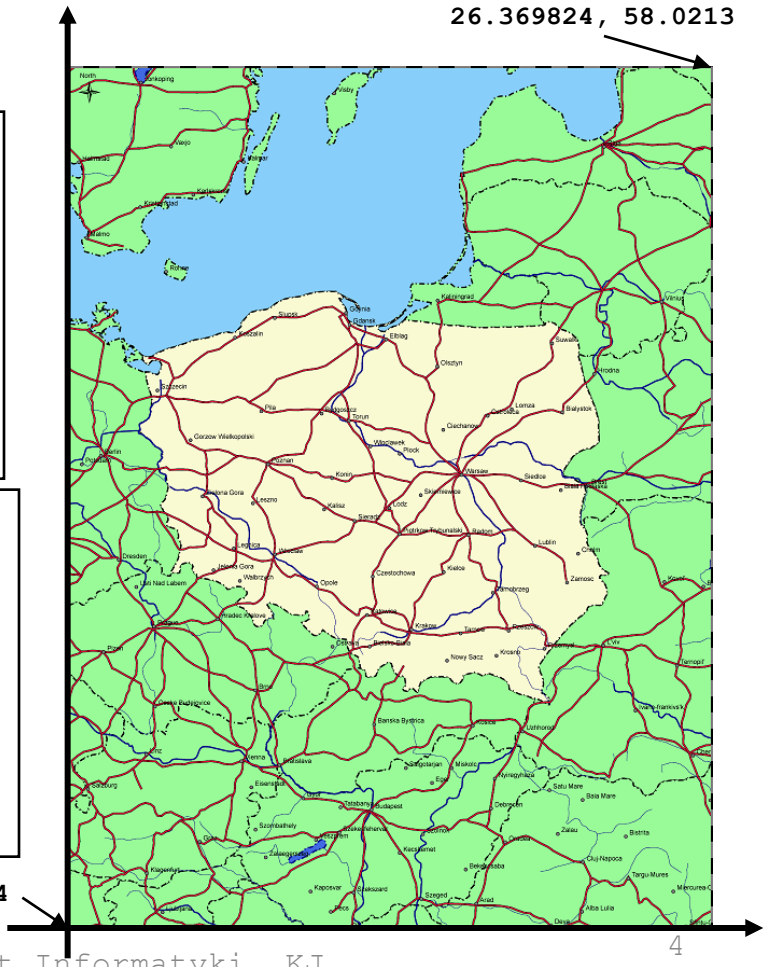
Nazwa kolumny	Typ kolumny	Opis
table_name	varchar2(32)	nazwa tabeli
column_name	varchar2(1024)	nazwa zarejestrowanej kolumny
diminfo	mdsys.sdo_dim_array	informacja dotycząca każdego z wykorzystywanych wymiarów; dla każdego wymiaru przechowywane są następujące dane: nazwa wymiaru, wartość wymiaru ograniczająca geometrie z dołu, wartość ograniczająca z góry, tolerancja.
srid	number	układ odniesienia wykorzystywany przez wszystkie geometrie w kolumnie

Metadane
wskazówki i przykład

- Rejestracja metadanych jest konieczna przed utworzeniem indeksu na kolumnie zawierającej geometrie
- Rejestracja jest realizowana tylko raz dla warstwy (czyli określonej kolumny w określonej tabeli)

```
INSERT INTO USER_SDO_GEOM_METADATA
VALUES (
'COUNTRY_BOUNDARIES',
'GEOM',
MDSYS.SDO_DIM_ARRAY(
  MDSYS.SDO_DIM_ELEMENT('X', 12.603676, 26.369824, 1),
  MDSYS.SDO_DIM_ELEMENT('Y', 45.8464, 58.0213, 1) ),
  8307
);
```

```
INSERT INTO USER_SDO_GEOM_METADATA
VALUES (
'MAJOR_CITIES',
'GEOM',
MDSYS.SDO_DIM_ARRAY(
  MDSYS.SDO_DIM_ELEMENT('X', 12.603676, 26.369824, 1),
  MDSYS.SDO_DIM_ELEMENT('Y', 45.8464, 58.0213, 1) ),
  8307
);
```

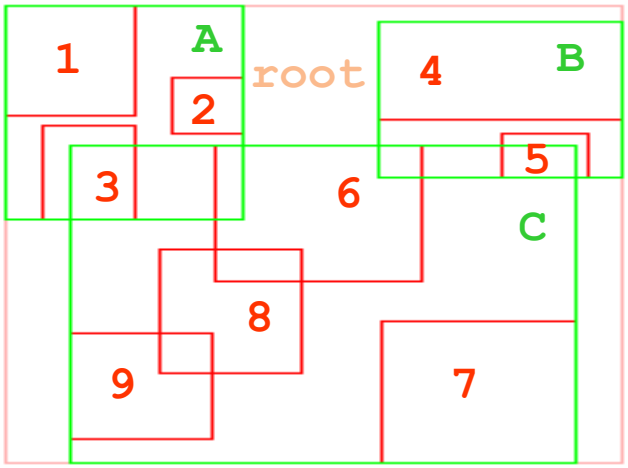


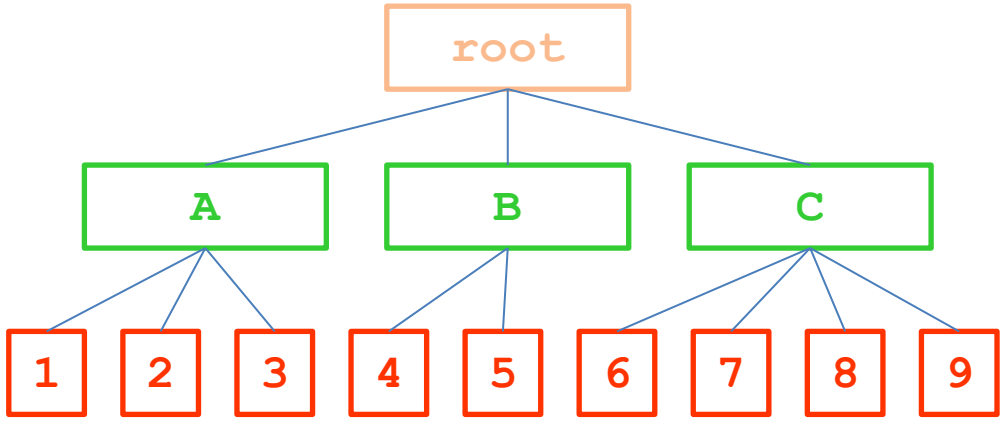
Indeksowanie danych przestrzennych

- Dane przestrzenne wymagają specyficznych technik indeksowania
- Do wersji 10g Oracle wspierał dwa typy indeksów: R-tree i Quadtree
- Od wersji 11g wspierany jest jedynie R-tree, w tej samej wersji pojawiło się pełne wsparcie indeksów (a co za tym idzie operatorów) dla obiektów 3D (parametr **`sdo_indx_dims=3`**)
- Od wersji 12.2 indeksy nie są już niezbędne podczas wyznaczania zależności pomiędzy geometriami, ale zalecane ze względów wydajnościowych

Własność	R-tree	Quadtree
Tworzenie i strojenie indeksu	Proste	Bardziej skomplikowane, dobranie odpowiednich wartości może mieć znaczące skutki
Wielkość indeksu	Mała	Duża
Zapytania z wyk. operatora najbliższego sąsiedztwa	Szybsza realizacja	Wolniejsza realizacja
Duża liczba modyfikacji	Zmniejszenie wydajności indeksu	Nie ma wpływu na wydajność
Zapytanie z wykorzystaniem SDO_WITHIN_DISTANCE	Zalecany	
Indeksowanych wymiarów	Do 4	Do 2
Przybliżanie geometrii	Słabe	Dobre przy odpowiednich właściwościach indeksu

Indeks R-tree (R-drzewo)





- Indeks R-tree dokonuje przybliżenia każdej geometrii przez zastosowanie pojedynczego najmniejszego obejmującego prostokąta (MBR – *minimum bounding rectangle*)
- Modyfikacje tabeli mogą wpływać na degradację jakości indeksu
  - Dawniej Jakość indeksu trzeba było ręcznie monitorować (podprogramami pakietu SDO_TUNE) i w razie potrzeby zlecić jego przebudowę
  - Obecnie system sam stroi indeks przestrzenny
- Przed utworzeniem indeksu można oszacować jego rozmiar w MB:
  - **`SDO_TUNE.ESTIMATE_RTREE_INDEX_SIZE`**

Indeks przestrzenny w Oracle

```
create index COUNTRY_BOUNDARIES_IDX
on COUNTRY_BOUNDARIES(GEOM)
INDEXTYPE IS MDSYS.SPATIAL_INDEX_V2;
```

- Oprócz udoskonalonej wersji indeksu przestrzennego (SPATIAL_INDEX_V2) nadal dostępny jest indeks w starszej wersji (SPATIAL_INDEX)
- Zalecane jest korzystanie z indeksu V2 ze względu na:
  - Kompletne wsparcie dla partycjonowania
  - Przewidywane przyszłe udoskonalenia
- Indeks V2 w dalszym ciągu domyślnie ma strukturę R-drzewa
- W przypadku indeksowania punktów 2D, można zlecić utworzenie indeksu V2 o strukturze B-drzewa
  - Lepsza wydajność operacji DML
  - Fizycznie struktura B-drzewa o 2-kolumnowym kluczu (x,y)

```
create index PT_IDX
on MY_2D_POINTS(GEOM)
INDEXTYPE IS MDSYS.SPATIAL_INDEX_V2
PARAMETERS ('layer_gtype=POINT cbtree_index=true');
```

Przykład podsumowujący (ciąg dalszy)

```
INSERT INTO USER_SDO_GEOM_METADATA
VALUES (
'geometrie1', 'geometria',
MDSYS.SDO_DIM_ARRAY(
MDSYS.SDO_DIM_ELEMENT('X', 0, 20, 0.005),
MDSYS.SDO_DIM_ELEMENT('Y', 0, 20, 0.005) ),
NULL );


CREATE INDEX figura_spatial_idx
ON geometrie1(geometria)
INDEXTYPE IS MDSYS.SPATIAL_INDEX_V2;
```

Przetwarzanie danych przestrzennych w praktyce

- Ładowanie danych przestrzennych
- Interfejs dla danych przestrzennych
  - Sposób przetwarzania zapytań
  - Metody podstawowe
  - Operatory
  - Funkcje jednowierszowe
  - Funkcje grupujące

Ładowanie danych przestrzennych

- Wiele narzędzi przetwarzających dane przestrzenne umożliwia konwersję danych przestrzennych – w tym konwersję do formatu danych przestrzennych w bazie danych Oracle.
- Załadowanie danych przestrzennych może być zrealizowane także za pomocą narzędzia SQLLoader.
- SQLLoader wymaga dokumentów zawierających dane do załadowania w określonym formacie
- Najczęstszym formatem dokumentów zawierających dane przestrzenne są tzw. *ShapeFile*
- Aby można było załadować pliki *ShapeFile* do wnętrza bazy danych Oracle, należy je przekonwertować.

Konwersja danych przestrzennych (1/3)

- ***Oracle Shapefile Converter*** – narzędzie ułatwiające konwersję danych z pliku *ShapeFile* do struktur w bazie danych Oracle

```
C:\>shp2sdo.exe MajorCities MAJOR_CITIES2
…
The following files have been created:
  MAJOR_CITIES2.sql :   SQL script to create the table
  MAJOR_CITIES2.ctl :   Control file for loading the table
  MAJOR_CITIES2.dat :   Data file

C:\>sqlplus scott/tiger @MAJOR_CITIES2.sql
…
Commit complete.
```

```
-- MAJOR_CITIES2.sql
DROP TABLE MAJOR_CITIES2;

CREATE TABLE MAJOR_CITIES2 (
  FIPS_CNTRY            VARCHAR2(2),
  CITY_NAME             VARCHAR2(40),
  GEOM      MDSYS.SDO_GEOMETRY);

DELETE FROM USER_SDO_GEOM_METADATA
  WHERE TABLE_NAME = 'MAJOR_CITIES2' AND COLUMN_NAME = 'GEOM' ;

INSERT INTO USER_SDO_GEOM_METADATA (TABLE_NAME, COLUMN_NAME, DIMINFO, SRID)
  VALUES ('MAJOR_CITIES2', 'GEOM',
    MDSYS.SDO_DIM_ARRAY
      (MDSYS.SDO_DIM_ELEMENT('X', 12.854999448, 26.316667448, 0.000000050),
       MDSYS.SDO_DIM_ELEMENT('Y', 45.868000183, 57.785999183, 0.000000050)
     ),
      NULL);
COMMIT;
```

Konwersja danych przestrzennych (2/3)

```
-- MAJOR_CITIES2.ctl
LOAD DATA
 INFILE MAJOR_CITIES2.dat
 TRUNCATE
 CONTINUEIF NEXT(1:1) = '#'
 INTO TABLE MAJOR_CITIES2
 FIELDS TERMINATED BY '|'
 TRAILING NULLCOLS (
   FIPS_CNTRY           NULLIF FIPS_CNTRY = BLANKS,
   CITY_NAME            NULLIF CNTRY_NAME = BLANKS,
   GEOM COLUMN OBJECT
   (
     SDO_GTYPE       INTEGER EXTERNAL,
     SDO_POINT COLUMN OBJECT
       (X            FLOAT EXTERNAL,
        Y            FLOAT EXTERNAL)
   )
)
```

```
-- MAJOR_CITIES2.dat
PL|Gdynia|2001|18,549999448|54,533298183|
PL|Slupsk|2001|17,033000448|54,462002183|
PL|Gdansk|2001|18,625000448|54,366001183|
PL|Koszalin|2001|16,184999448|54,186001183|
PL|Elblag|2001|19,405000448|54,159000183|
PL|Suwalki|2001|22,940000448|54,103000183|
PL|Olsztyn|2001|20,492000448|53,778000183|
PL|Szczecin|2001|14,531000448|53,438000183|
PL|Lomza|2001|22,080999448|53,172001183|
PL|Pila|2001|16,745000448|53,145000183|
PL|Bialystok|2001|23,159000448|53,130001183|
PL|Bydgoszcz|2001|18,028999448|53,111999183|
PL|Ostroleca|2001|21,568000448|53,080002183|
PL|Torun|2001|18,611000448|53,011001183|
PL|Ciechanow|2001|20,621999448|52,882000183|
PL|Gorzow Wielkopolski|2001|15,236000448|52,736000183|
PL|Wloclawek|2001|19,066999448|52,646999183|
PL|Plock|2001|19,691999448|52,544998183|
. . .
```

```
C:\>sqlldr scott/tiger MAJOR_CITIES2

SQL*Loader: Release 10.2.0.1.0 - Production on Cz Maj 15 15:36:25 2008
Copyright (c) 1982, 2005, Oracle.  All rights reserved.
Commit point reached - logical record count 62
```

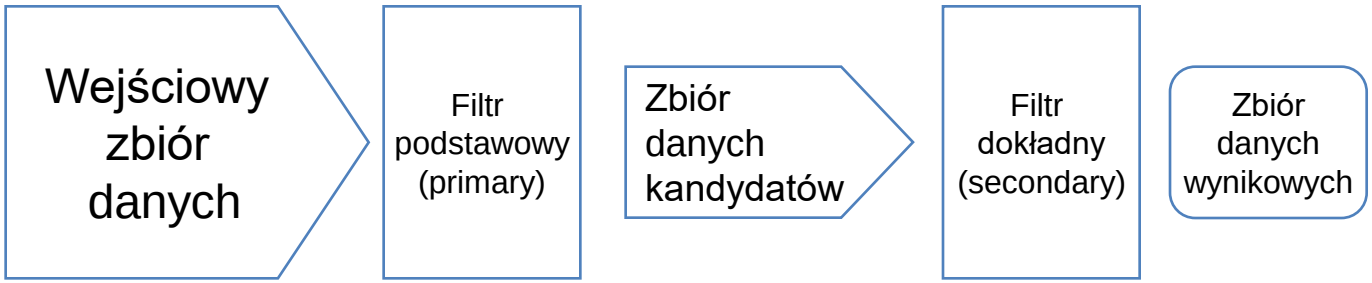
Konwersja danych przestrzennych (3/3)

- Aby można było wykorzystać załadowane dane ze starszych wersji Oracle Spatial należy dokonać konwersji ich zawartości typu **SDO_GEOMETRY do** zgodnej z obecnie obowiązującymi regułami.
- Przykładowo, we wcześniejszych wersjach nie były wykorzystywane zewnętrzne i wewnętrzne wielokąty (wykorzystywany był jedynie typ 3).

```
C:\>sqlplus scott/tiger

SQL> create index MAJOR_CITIES2_IDX on MAJOR_CITIES2(GEOM)
  2  INDEXTYPE IS MDSYS.SPATIAL_INDEX;

Index created.

SQL> EXECUTE SDO_MIGRATE.TO_CURRENT('MAJOR_CITIES2','GEOM');

PL/SQL procedure successfully completed.
```

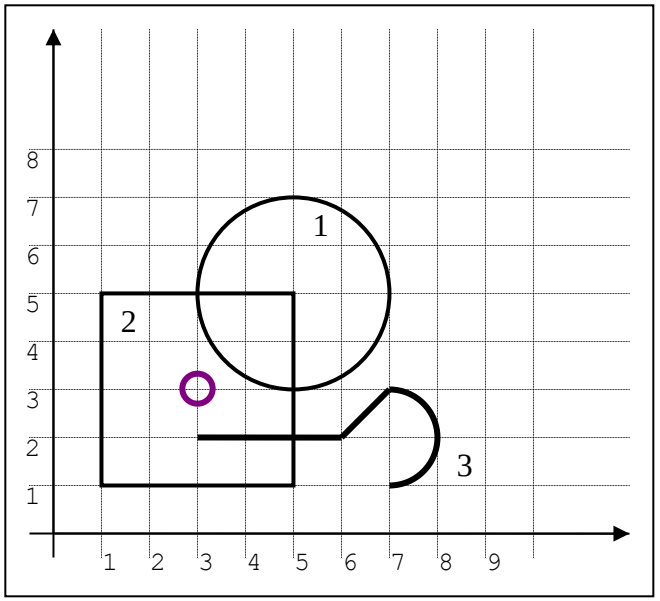
Sposób przetwarzania zapytań

- Zapytania przestrzenne wykorzystują dwufazowe przetwarzanie.
- Podczas pierwszej fazy z pełnego zbioru danych na podstawie aproksymacji opartej na indeksie przestrzennym wybierane są te obiekty, które potencjalnie mogą spełniać warunki zapytania są to tzw. kandydaci.
- W fazie drugiej ze zbioru kandydatów wyznaczane są za pomocą metod dokładnych te obiekty, które rzeczywiście spełniają zadany warunek.



Sposób przetwarzania zapytań przykład

- Operator **SDO_FILTER**, który wykorzystuje jedynie pierwszą fazę zapytania, czyli daje w wyniku zbiór "kandydatów", dla indeksu r-tree uzna, że z punktem 3,3 mają "coś wspólnego" wszystkie 3 geometrie.
- Operator **SDO_RELATE**, który wykorzystuje obie fazy zapytania, czyli dokonuje dodatkowej weryfikacji zbioru kandydatów, będzie miał na ten temat "odmienne zdanie".




```
select ID
from FIGURY
where SDO_FILTER(KSZTALT,
SDO_GEOMETRY(2001,null,
          SDO_POINT_TYPE(3,3,null),
          null,null)) = 'TRUE';
 ID
----------
          3
          2
          1
```

```
select ID
from FIGURY
where SDO_RELATE(KSZTALT,
        SDO_GEOMETRY(2001,null,
          SDO_POINT_TYPE(3,3,null),
          null,null),
        'mask=ANYINTERACT') = 'TRUE';
ID
----------
          2
```

Podstawowe metody typu SDO_GEOMETRY

- Jak większość wbudowanych typów obiektowych **SDO_GEOMETRY** oprócz atrybutów posiada również metody.
- Wszystkie metody typu **SDO_GEOMETRY** są funkcjami.
- Do najbardziej podstawowych metod należą:
  - **GET_GTYPE** – daje w wyniku typ geometrii (*tt*),
  - **GET_DIMS** – liczba wymiarów geometrii (*d*),
  - **GET_LRS_DIM** – pozycja miary związanej z LRS wśród współrzędnych (*l*).

```
select T.CNTRY_NAME, T.GEOM.GET_GTYPE(), T.GEOM.GET_DIMS(), T.GEOM.GET_LRS_DIM()
from   COUNTRY_BOUNDARIES T
```

CNTRY_NAME	T.GEOM.GET_GTYPE()	T.GEOM.GET_DIMS()	T.GEOM.GET_LRS_DIM()
Poland	3	2	0
Czech Republic	3	2	0
Germany	7	2	0
Slovakia	3	2	0
Russia	7	2	0
Estonia	7	2	0
Sweden	7	2	0
…			

Operatory i funkcje

- Dane przestrzenne można przetwarzać za pomocą bardzo bogatego zbioru operatorów i funkcji.
- Podział ich jest następujący:
  - Operatory przestrzenne (*Spatial Operators*)
  - Funkcje geometryczne (*Geometry Functions*)
  - Przestrzenne funkcje grupujące (*Spatial Aggregate Functions*)

Operatory przestrzenne

- **SDO_FILTER** – sprawdza czy dwa obiekty występują w określonej zależności. Wykorzystuje tylko filtr podstawowy, wykorzystuje indeks.
- **SDO_NN** – służy do identyfikacji najbliższego sąsiedztwa, wykorzystuje indeks.
- **SDO_NN_DISTANCE** – zwraca odległość od obiektów zwróconych przez operator **SDO_NN**.
- **SDO_RELATE** – wyznacza obiekty będące w określonej zależności przestrzennej z geometrią, wykorzystuje indeks.
- **SDO_WITHIN_DISTANCE** – wyznacza zbiór geometrii znajdujących się w ramach określonej odległości od obiektu, wykorzystuje indeks.
- W przypadku każdego z powyższych operatorów, kolumna na której został założony indeks R-tree musi być pierwszym argumentem.
- **SDO_JOIN** – wykonuje połączenie przestrzenne (spatial join) dwóch zbiorów geometrii zgodnie z podaną zależnością przestrzenną, wymaga indeksów na kolumnach zawierających oba zbiory.

Funkcje geometryczne (1/2)

- Funkcje geometryczne można pogrupować następująco:
  - Wyznaczające relacje pomiędzy dwoma obiektami:
    - **`RELATE`** – określa zależność pomiędzy obiektami,
    - **`WITHIN_DISTANCE`** – sprawdza czy obiekty znajdują się w określonej odległości
  - Walidujące:
    - **`VALIDATE_GEOMETRY`** – sprawdza poprawność geometrii,
    - **`VALIDATE_LAYER`** – sprawdza poprawność warstwy (wszystkich geometrii w kolumnie określonej tabeli)
  - Operacje na dwóch obiektach:
    - **`SDO_DISTANCE`** – oblicza odległość między geometriami,
    - **`SDO_DIFFERENCE`** – różnica topologiczna dwóch geometrii,
    - **`SDO_INTERSECTION`** – część wspólna dwóch geometrii,
    - **`SDO_UNION`** – suma topologiczna dwóch geometrii,
    - **`SDO_XOR`** – symetryczna różnica dwóch geometrii

Funkcje geometryczne (2/2)

- Operacje na pojedynczych obiektach:
  - **SDO_ARC_DENSIFY** – zamienia łuki i okręgi na przybliżone odpowiedniki wyrażone za pomocą linii prostych,
  - **SDO_AREA** – wyznacza powierzchnię dwuwymiarowej geometrii,
  - **SDO_BUFFER** – generuje wielokąt otaczający geometrię,
  - **SDO_CENTROID** – zwraca punkt georeferencyjny obrębu (tzw. controid) geometrii,
  - **SDO_CONVEXHULL** – zwraca obiekt reprezentujący powłokę wypukłą (convex hull) geometrii,
  - **SDO_LENGTH** – wylicza długość obwodu,
  - **SDO_MBR** – wyznacza najmniejszy prostokąt obejmujący geometrię,
  - **SDO_[MIN|MAX]_MBR_ORDINATE** – wyznacza "dolną" ("górną") krawędź MBR obejmującego geometrię dla określonego wymiaru,
  - **SDO_POINTONSURFACE** – zwraca punkt należący do powierzchni geometrii

Przestrzenne funkcje grupujące

- **SDO_AGGR_CENTROID*** – zwraca punkt georeferencyjny (środek ciężkości) zbioru geometrii
- **SDO_AGGR_CONVEXHULL*** – obiekt reprezentujący powłokę wypukłą (convex hull) zbioru geometrii,
- **SDO_AGGR_LRS_CONCAT*** – wyznacza geometrię LRS będącą konkatenacją zbioru geometrii LRS
- **SDO_AGGR_MBR** – wyznacza najmniejszy prostokąt obejmujący zbiór geometrii
- **SDO_AGGR_UNION*** – wyznacza topologiczną sumę zbioru geometrii

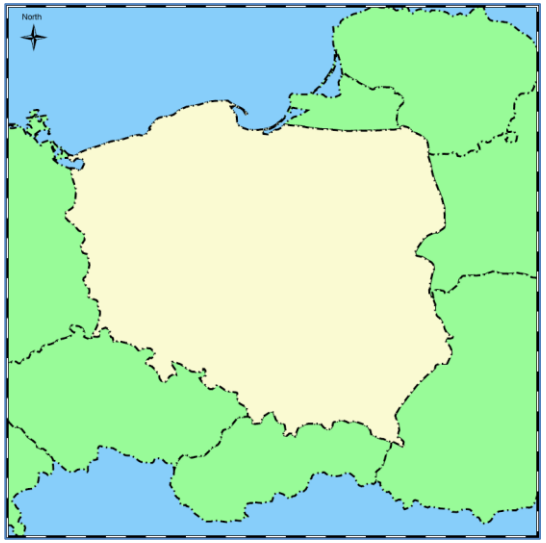
* - Większość funkcji agregujących akceptuje parametr typu **MDSYS.SDOAGGRTYPE** a nie bezpośrednio **MDSYS.SDO_GEOMETRY**.
(**SDOAGGRTYPE = SDO_GEOMETRY + tolerance**)

Operatory przestrzenne
SDO_FILTER

```
SDO_FILTER( geometria1, geometria2 [, parametry])
```

```
select A.CNTRY_NAME A_NAME, B.CNTRY_NAME B_NAME
from   COUNTRY_BOUNDARIES A, COUNTRY_BOUNDARIES B
where  SDO_FILTER(A.GEOM, B.GEOM) = 'TRUE'
and    B.CNTRY_NAME = 'Poland';
```

A_NAME	B_NAME
Lithuania	Poland
Byelarus	Poland
Russia	Poland
Poland	Poland
Ukraine	Poland
Germany	Poland
Slovakia	Poland
Czech Republic	Poland



Operatory przestrzenne
SDO_NN i SDO_NN_DISTANCE

`SDO_NN(geometria1, geometria2, parametry [, numer])`

`SDO_NN_DISTANCE(numer)`

- Parametr **sdo_num_res** operatora **SDO_NN** określa liczbę obiektów z najbliższego sąsiedztwa jaką należy zwrócić w zapytaniu.
- Parametr **unit** operatora **SDO_NN** pozwala wyrazić odległość przez operator **SDO_NN_DISTANCE** w pożądanych jednostkach.
- Jedyny parametr operatora **SDO_NN_DISTANCE** musi być liczbą podaną w operatorze **SDO_NN** jako ostatni parametr.

```
select A.CITY_NAME, ROUND(SDO_NN_DISTANCE(1)) DISTANCE
from   MAJOR_CITIES A
where  SDO_NN(GEOM,MDSYS.SDO_GEOMETRY(2001, 8307, NULL,
                         MDSYS.SDO_ELEM_INFO_ARRAY(1, 1, 1),
                         MDSYS.SDO_ORDINATE_ARRAY(16.91673,52.38237)),
               'sdo_num_res=4 unit=km',1) = 'TRUE';
```

CITY_NAME	DISTANCE
Pila	86
Poznan	3
Konin	93
Leszno	65

Operatory przestrzenne
SDO_RELATE

```
SDO_RELATE(geometria1, geometria2, parametry)
```

- W operatorze **SDO_RELATE** podstawowym parametrem jest:
  - **mask** – jedna z dziewięciu zależności geometrycznych: **TOUCH**, **OVERLAPBDYDISJOINT**, **OVERLAPBDYINTERSECT**, **EQUAL**, **INSIDE**, **COVEREDBY**, **CONTAINS**, **COVERS**, **ANYINTERACT**, **ON**. Jeśli podano kilka masek, to są one łączone logicznym operatorem **OR** np: '**mask=inside+touch**'

```
select B.CNTRY_NAME, count(*)
from   COUNTRY_BOUNDARIES B, MAJOR_CITIES C
where  SDO_RELATE(C.GEOM, B.GEOM,
       'mask=INSIDE') = 'TRUE'
group  by B.CNTRY_NAME;
```

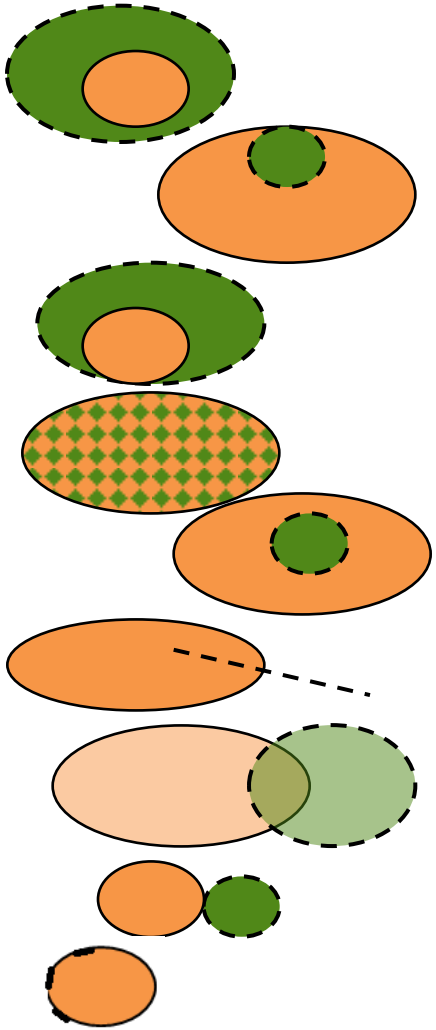
CNTRY_NAME	COUNT(*)
Poland	50
Slovakia	3
Denmark	1
Slovenia	1
Austria	6
Germany	3
Byelarus	2
Hungary	19
Latvia	1
Romania	13
Czech Republic	7
Russia	1
Sweden	7
Ukraine	7
Lithuania	1

- Zamiast operatora **SDO_RELATE** z odpowiednią maską można użyć jednego z dedykowanych operatorów: **SDO_TOUCH**, **SDO_OVERLAPBDYDISJOINT**, **SDO_OVERLAPBDYINTERSECT**, **SDO_EQUAL**, **SDO_INSIDE**, **SDO_COVEREDBY**, **SDO_CONTAINS**, **SDO_COVERS**, **SDO_ANYINTERACT**, **SDO_ON**.

Zależności pomiędzy geometriami

SDO_RELATE(geometria1, geometria2,...)

- **ANYINTERACT** – wówczas gdy obiekty nie są rozłączne.
- **CONTAINS** – jeśli drugi obiekt całkowicie zawiera się wewnątrz pierwszego obiektu, granice obiektów się nie stykają
- **COVEREDBY** – jeśli pierwszy obiekt jest całkowicie zawarty wewnątrz drugiego obiektu, granice obiektów stykają się w jednym lub wielu punktach.
- **COVERS** – jeśli drugi obiekt jest całkowicie wewnątrz pierwszego obiektu, granice nakładają się w jednym lub wielu miejscach.
- **EQUAL** – jeśli obiekty współdzielą każdy punkt swoich granic i wnętrza włączając w to dziury wewnątrz obiektów.
- **INSIDE** – jeśli pierwszy obiekt całkowicie zawiera się wewnątrz drugiego obiektu, granice nie stykają się.
- **OVERLAPBDYDISJOINT** – jeśli obiekty zachodzą na siebie lecz ich granice nie mają części wspólnych.
- **OVERLAPBDYINTERSECT** – jeśli obiekty zachodzą na siebie a ich granice mają części wspólne w jednym lub wielu fragmentach.
- **TOUCH** – jeśli dwa obiekty współdzielą wspólne punkty graniczne i jednocześnie nie współdzielą żadnych punktów wewnętrznych.
- **ON** – jeśli wnętrze i granica jednego obiektu zawierają się w granicy drugiego obiektu.



Operatory przestrzenne
SDO_WITHIN_DISTANCE

```
SDO_WITHIN_DISTANCE(geometria1, geometria2, parametry);
```

- Podstawowe parametry operatora **SDO_WITHIN_DISTANCE**:
  - **distance** – odległość od geometria2
  - **querytype** – deklaracja dotycząca wykorzystania filtru dokładnego. Wartość równa **FILTER** powoduje wykorzystanie tylko filtru podstawowego

```
select C.CITY_NAME
from   MAJOR_CITIES C
where  SDO_WITHIN_DISTANCE(C.GEOM,
                           SDO_GEOMETRY(2001,
                                        8307,
                                        null,
                                        MDSYS.SDO_ELEM_INFO_ARRAY(1, 1, 1),
                                        MDSYS.SDO_ORDINATE_ARRAY(16.91673, 52.38237)),
                           'distance=100 unit=km') = 'TRUE';
```

CITY_NAME
Pila
Poznan
Konin
Leszno

Funkcje geometryczne
SDO_GEOM.RELATE

```
SDO_GEOM.RELATE(
  geometria1 MDSYS.SDO_GEOMETRY, mask  VARCHAR2,
  geometria2 MDSYS.SDO_GEOMETRY, tolerancja NUMBER ) RETURN VARCHAR2;
```

- Wyniki działania funkcji **SDO_GEOM.RELATE** mogą być następujące:
  - Jeśli w parametrze **mask** określono oczekiwany związek pomiędzy obiektami, wówczas funkcja zwraca ciąg reprezentujący ten związek lub ciąg znaków '**FALSE**'
  - Jeśli w parametrze **mask** został podany ciąg '**DETERMINE**' wówczas funkcja daje w wyniku ciąg reprezentujący związek pomiędzy geometriami
  - W przypadku **mask** równego '**ANYINTERACT**' funkcja daje w wyniku '**TRUE**' pod warunkiem, że mamy do czynienia z nierozłącznymi geometriami

CNTRY	NAME	RELATION
Poland	Nogat	CONTAINS
Poland	Vistula	CONTAINS
Poland	Odra	CONTAINS
Poland	Bug	CONTAINS
Poland	Dniester	DISJOINT
Poland	Oder-Spree-Kanal	DISJOINT
Poland	Vistula	OVERLAPBDYDISJOINT
Poland	San	OVERLAPBDYDISJOINT
Poland	Odra	OVERLAPBDYINTERSECT
Poland	Morava	OVERLAPBDYINTERSECT
Poland	Oder	OVERLAPBDYINTERSECT
Poland	Oder-Havel-Kanal	TOUCH

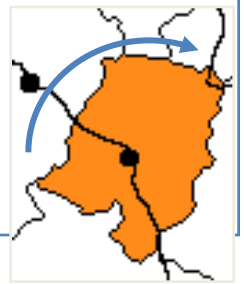
```
select distinct B.CNTRY_NAME, R.name,
SDO_GEOM.RELATE(B.GEOM, 'DETERMINE', R.GEOM, 1) RELATION
from   COUNTRY_BOUNDARIES B, RIVERS R
where  B.CNTRY_NAME = 'Poland'
```

Funkcje geometryczne
SDO_GEOM.VALIDATE_GEOMETRY

```
SDO_GEOM.VALIDATE_GEOMETRY(
  geometria MDSYS.SDO_GEOMETRY, tolerancja NUMBER ) RETURN VARCHAR2;
```

- Funkcja weryfikująca spójność geometrii.
- Sprawdza reprezentację geometrii na podstawie definicji elementu
- Zalecanym jej odpowiednikiem jest funkcja **GEOM.VALIDATE_GEOMETRY_WITH_CONTEXT**
- Wyniki działania funkcji są następujące:
  - **TRUE** – jeśli geometria jest prawidłowa
  - Jeśli posiada błędy wówczas uzyskujemy numer błędu Oracle'a określający przyczynę błędu
  - W przypadku funkcji **GEOM.VALIDATE_GEOMETRY_WITH_CONTEXT** otrzymujemy dodatkowo kontekst błędu – informacje w jakim miejscu definicji geometrii błąd został znaleziony
- Funkcja weryfikuje zarówno spójność typu jak i spójność geometrii

```
SQL> select SDO_GEOM.VALIDATE_GEOMETRY_WITH_CONTEXT(WO_KSZTALT,0.01) VALID,
  2         WO_NAZWA from SO_WOJEWODZTWA
  3  where SDO_GEOM.VALIDATE_GEOMETRY(WO_KSZTALT,0.01) <> 'TRUE';
VALID                                          WO_NAZWA
---------------------------------------------- --------------------
13367 [Element <1>] [Ring <1>]                 opolskie
```



ORA-13367 Wrong orientation for interior/exterior rings

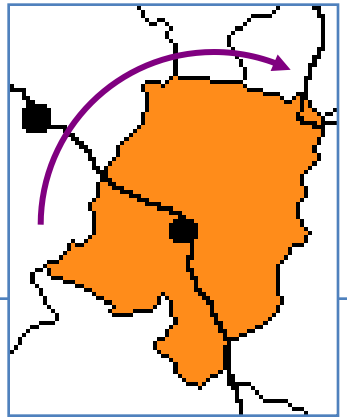
Funkcje geometryczne
SDO_GEOM.VALIDATE_LAYER

```
SDO_GEOM.VALIDATE_LAYER_WITH_CONTEXT(
  tabela VARCHAR2, kolumna_geom VARCHAR2,
  tabela_wynikowa VARCHAR2[, częstość_zatwierdzania NUMBER]);
```

- Zalecanym odpowiednikiem jest **SDO_GEOM.VALIDATE_LAYER_WITH_CONTEXT**
- Procedura wypełnia tabelę wynikową informacjami o rezultatach walidacji wszystkich geometrii w warstwie
- Tabela wynikowa powinna być utworzona przed wywołaniem procedury

```
CREATE TABLE val_results (sdo_rowid ROWID, result varchar2(1000));

EXECUTE SDO_GEOM.VALIDATE_LAYER_WITH_CONTEXT('SO_WOJEWODZTWA','WO_KSZTALT','VAL_RESULTS');
PL/SQL procedure successfully completed.

SQL> SELECT * from val_results;

SDO_ROWID          RESULT
------------------ ----------------------------------------
                   Rows Processed <16>
AAAKl7AABAAANMyAAD 13367 [Element <1>] [Ring <1>]
```

ORA-**13367** Wrong orientation for interior/exterior rings



Zakres walidacji geometrii

- Walidacja geometrii obejmuje dwa typy weryfikacji
  - Weryfikację **typu** – odbywa się na etapie wstawiania obiektu do bazy danych
  - Weryfikację **spójności geometrii** – możliwa za pomocą przeznaczonych do tego funkcji
- Weryfikacja typu
  - Prawidłowa wartość **`SDO_GTYPE`**
  - Wartości **`SDO_ETYPE`** muszą być zgodne z **`SDO_GTYPE`**
  - Czy **`SDO_ELEM_INFO_ARRAY`** posiada liczbę wartości podzielną przez trzy

Zakres walidacji geometrii

- Weryfikacja spójności geometrii:
  - Wielokąt musi posiadać co najmniej cztery punkty
  - Wielokąt musi być domknięty
  - Żadne dwa punkty w linii lub wielokącie nie są takie same
  - Wielokąt musi być opisany w odpowiednim kierunku (zewnętrzna krawędź odwrotnie, wewnętrzna zgodnie z ruchem wskazówek zegara)
  - Wewnętrzny wielokąt nie może dotykać więcej niż raz wielokąta zewnętrznego
  - Wewnętrzne wielokąty nie mogą się stykać więcej niż jednym punktem
  - Ciąg linii zawiera co najmniej dwa punkty
  - Jedno i czterocyfrowe wartości **`SDO_ETYPE`** nie mogą być wymieszane przy definiowaniu wielokątów
  - Punkty w przypadku koła lub łuku nie mogą znajdować się na linii prostej
  - Geometrie muszą znajdować się w granicach wyznaczonych przez metadane
  - Geometrie LRS posiadają trzy lub cztery wymiary oraz prawidłową pozycję liczby wymiarów

Funkcje geometryczne

SDO_GEOM.SDO_DISTANCE

```
SDO_GEOM.SDO_DISTANCE(
geometria1 MDSYS.SDO_GEOMETRY, geometria2 MDSYS.SDO_GEOMETRY,
tolerancja IN NUMBER  [, jednostki IN VARCHAR2]) RETURN NUMBER;
```

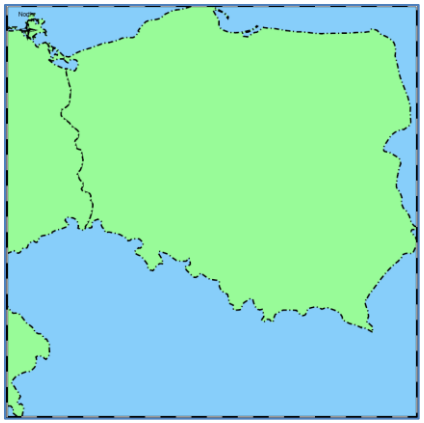
```
select A.CITY_NAME, B.CITY_NAME,
       ROUND(SDO_GEOM.SDO_DISTANCE(A.GEOM, B.GEOM, 1, 'unit=km')) ODL
from   MAJOR_CITIES A, MAJOR_CITIES B
where  A.CITY_NAME = 'Poznan'
and    B.CITY_NAME in ('Berlin', 'Warsaw');
```

CITY_NAME	CITY_NAME	ODL
Poznan	Berlin	242
Poznan	Warsaw	281

SDO_GEOM.SDO_INTERSECTION

```
SDO_GEOM.SDO_INTERSECTION(
geometria1 MDSYS.SDO_GEOMETRY, geometria2 MDSYS.SDO_GEOMETRY,
tolerancja NUMBER ) RETURN NUMBER;
```

```
select SDO_GEOM.SDO_INTERSECTION(A.GEOM, B.GEOM, 1) GRANICA
from   COUNTRY_BOUNDARIES A, COUNTRY_BOUNDARIES B
where  A.CNTRY_NAME = 'Poland'
and    B.CNTRY_NAME = 'Germany';
```

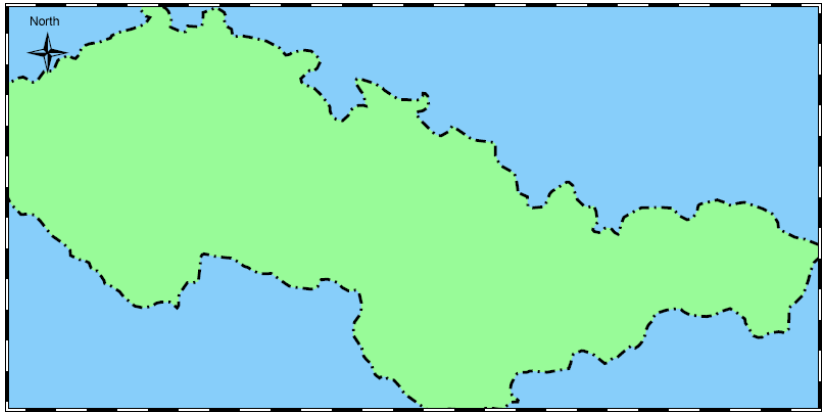


```
SDO_GEOMETRY(2006, 8307, NULL,
   SDO_ELEM_INFO_ARRAY(1, 2, 1, 125, 2, 1),
   SDO_ORDINATE_ARRAY(14.275627, 53.699066, 14.309721, 53.55555, 14.413261, 53.33 8959, …) )
```

Funkcje geometryczne
SDO_GEOM.SDO_UNION

```
SDO_GEOM.SDO_UNION(
geometria1 MDSYS.SDO_GEOMETRY, geometria2 MDSYS.SDO_GEOMETRY,
tolerancja IN NUMBER ) RETURN MDSYS.SDO_GEOMETRY;
```

```
select SDO_GEOM.SDO_UNION(A.GEOM, B.GEOM, 1) CZECHOSLOWACJA
from   COUNTRY_BOUNDARIES A, COUNTRY_BOUNDARIES B
where  A.CNTRY_NAME = 'Czech Republic'
and    B.CNTRY_NAME = 'Slovakia';

CZECHOSLOWACJA
----------------------------------------------------
SDO_GEOMETRY(2003, 8307, NULL,
   SDO_ELEM_INFO_ARRAY(1, 1003, 1),
   SDO_ORDINATE_ARRAY(18.851246, 49.517357, 18.852219, 49.527771, 18.847775, 49.554161,
                      18.839443, 49.594994, 18.810831, 49.673328, 18.786942, 49.681938, …) )
```



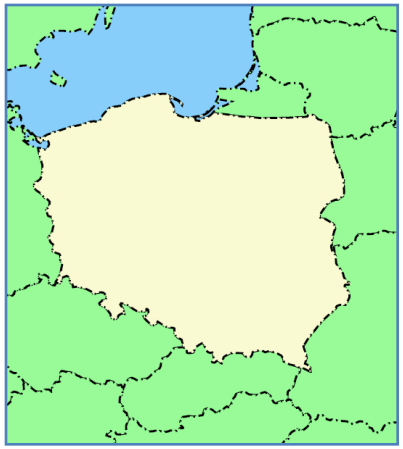
Funkcje geometryczne

SDO_GEOM.SDO_AREA

```
SDO_GEOM.SDO_AREA(
  geometria MDSYS.SDO_GEOMETRY,
  tolerancja NUMBER[, jednostki VARCHAR2] ) RETURN NUMBER;
```

```
select A.CNTRY_NAME,
       ROUND(SDO_GEOM.sdo_area(A.GEOM, 1, 'unit=SQ_KM')) POWIERZCHNIA
from   COUNTRY_BOUNDARIES A
order by 2 desc
```

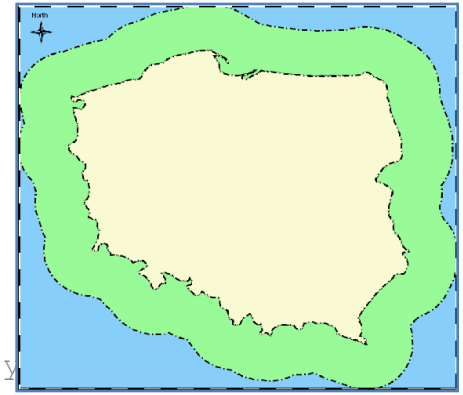
CNTRY_NAME	POWIERZCHNIA
Poland	311664
Ukraine	100551
Hungary	92498
Romania	84721
Czech Republic	77447
Austria	68551



SDO_GEOM.SDO_BUFFER

```
SDO_GEOM.SDO_BUFFER(
geometria MDSYS.SDO_GEOMETRY, odległość NUMBER,
tolerancja NUMBER[, parametry VARCHAR2] ) RETURN MDSYS.SDO_GEOMETRY;
```

```
select SDO_GEOM.SDO_BUFFER(A.GEOM, 100, 1, 'unit=km') GEOM
from   COUNTRY_BOUNDARIES A
where  A.CNTRY_NAME = 'Poland'
```



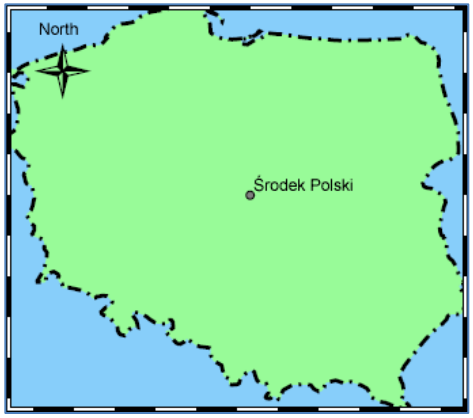
Funkcje geometryczne

SDO_GEOM.SDO_CENTROID

```
SDO_GEOM.SDO_CENTROID(
geometria MDSYS.SDO_GEOMETRY, tolerancja NUMBER) RETURN MDSYS.SDO_GEOMETRY;
```

```
select SDO_GEOM.SDO_CENTROID(B.GEOM,1) SRODEK_POLSKI
from   COUNTRY_BOUNDARIES B
where  B.CNTRY_NAME = 'Poland';

SRODEK POLSKI
------------------------------
SDO_GEOMETRY(2001, 8307,
    SDO_POINT_TYPE(19.4302985, 52.1100187, NULL), NULL, NULL)
```

Pomiędzy Goślubem a Siemieniczkami, na południe od Kutna



SDO_GEOM.SDO_LENGTH

```
SDO_GEOM.SDO_LENGTH( geometria MDSYS.SDO_GEOMETRY,
tolerancja NUMBER [, jednostki VARCHAR2]) RETURN NUMBER;
```

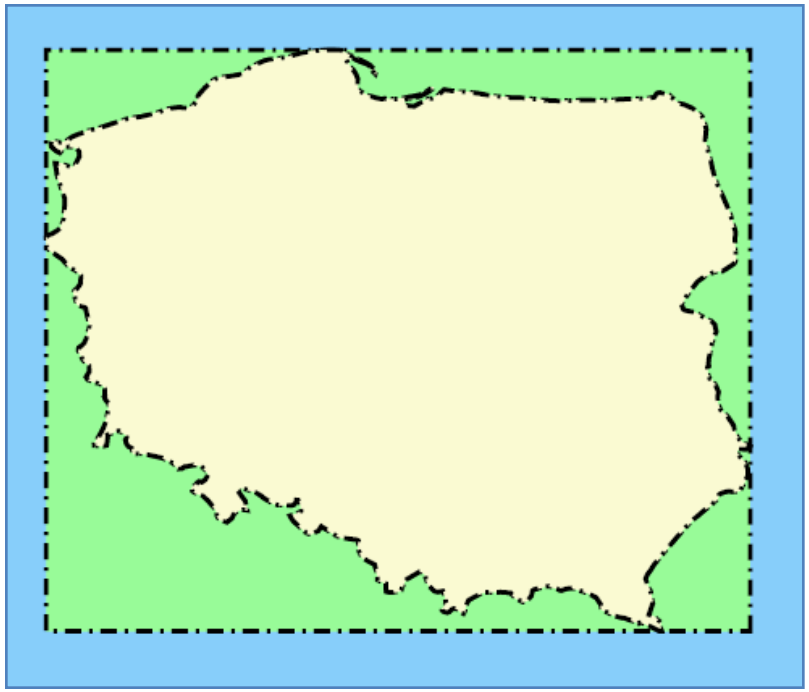
```
select A.CNTRY_NAME,
       B.CNTRY_NAME,
       ROUND(SDO_GEOM.SDO_LENGTH(SDO_GEOM.SDO_INTERSECTION(A.GEOM, B.GEOM, 1), 1, 'unit=km'))
from   COUNTRY_BOUNDARIES A,
       COUNTRY_BOUNDARIES B
where  A.CNTRY_NAME = 'Poland';
```

CNTRY_NAME	CNTRY_NAME	DLUGOSCI_GRANIC
Poland	Lithuania	82
Poland	Russia	197
Poland	Byelarus	322
Poland	Slovakia	374
Poland	Germany	376
Poland	Ukraine	391
Poland	Czech Republic	525

Funkcje geometryczne
SDO_GEOM.SDO_MBR

```
SDO_GEOM.SDO_MBR( geometria MDSYS.SDO_GEOMETRY
[, definicje_wymiarów IN MDSYS.SDO_DIM_ARRAY] ) RETURN MDSYS.SDO_GEOMETRY;
```

```
select SDO_GEOM.SDO_MBR(A.GEOM) GEOM
from   COUNTRY_BOUNDARIES A
where  A.CNTRY_NAME = 'Poland'
--------------------------------------------------------------------------------
SDO_GEOMETRY(2003, NULL, NULL, SDO_ELEM_INFO_ARRAY(1, 1003, 3), SDO_ORDINATE_ARR
AY(14.147637, 49.002914, 24.143469, 54.836037))
```

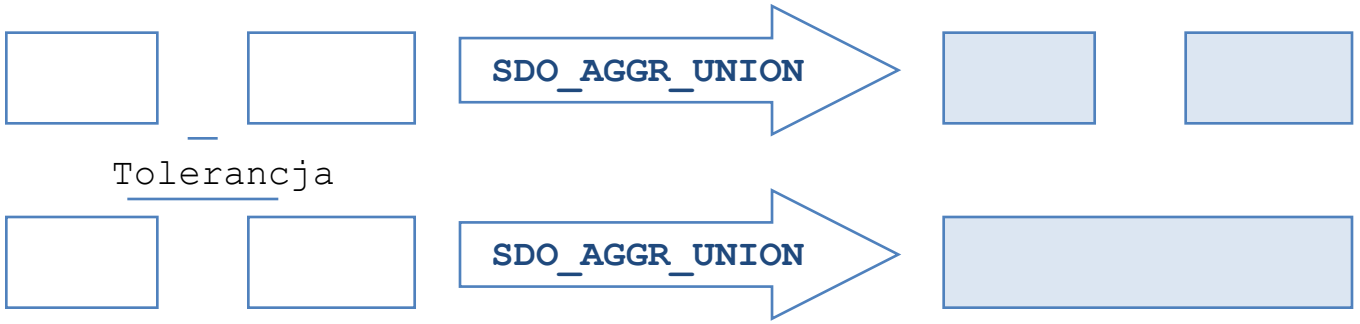


Przestrzenne funkcje grupujące
Typ **MDSYS.SDOAGGRTYPE**

- Wiele funkcji agregujących przyjmuje jako parametr obiekt typu **MDSYS.SDOAGGRTYPE**
- Definicja typu **MDSYS.SDOAGGRTYPE** jest następująca:

```
CREATE TYPE sdoaggrtype AS OBJECT (
  geometria MDSYS.SDO_GEOMETRY,
  tolerancja NUMBER);
```

- Tolerancja ma wpływ na wyniki działania wielu funkcji przestrzennych, w tym funkcji agregujących. Definiuje ona odległość, która ma być "pomijana" w obliczeniach



Przestrzenne funkcje grupujące
SDO_AGGR_UNION

```
SDO_AGGR_UNION(
AggregateGeometry MDSYS.SDOAGGRTYPE) RETURN MDSYS.SDO_GEOMETRY;
```

```
select SDO_AGGR_UNION(MDSYS.SDOAGGRTYPE(B.GEOM,1)) GEOM
from   COUNTRY_BOUNDARIES B
```

```
SDO_GEOMETRY(2007, 8307, NULL,
   SDO_ELEM_INFO_ARRAY(1, 1003, 1, 31, 1003, 1, 219, 1003, 1, 269, 1003, 1, 277, 1003, 1,
                       381, 1003, 1, 703, 1003, 1, 805, 1003, 1, 843, 1003, 1, 865, 1003, 1),
   SDO_ORDINATE_ARRAY(12.900692, 54.442356, 12.803333, 54.442497, 12.680777, 54.444942,
                      12.660443, 54.44611, 12.63 8611, 54.448944, 12.603676, 54.452486, …) )
```