geomatika FAV

KGM/PDB
Prostorové objekty a SQL

Karel Janečka

Tvorba materiálů byla podpořena z prostředků projektu FRVŠ č. F0584/2011/F1d

- Norma ISO 19125
  - Geometrický model OGC
  - Implementace geometrického modelu OGC v prostředí SQL
- Použití ADT pro prostorové objekty

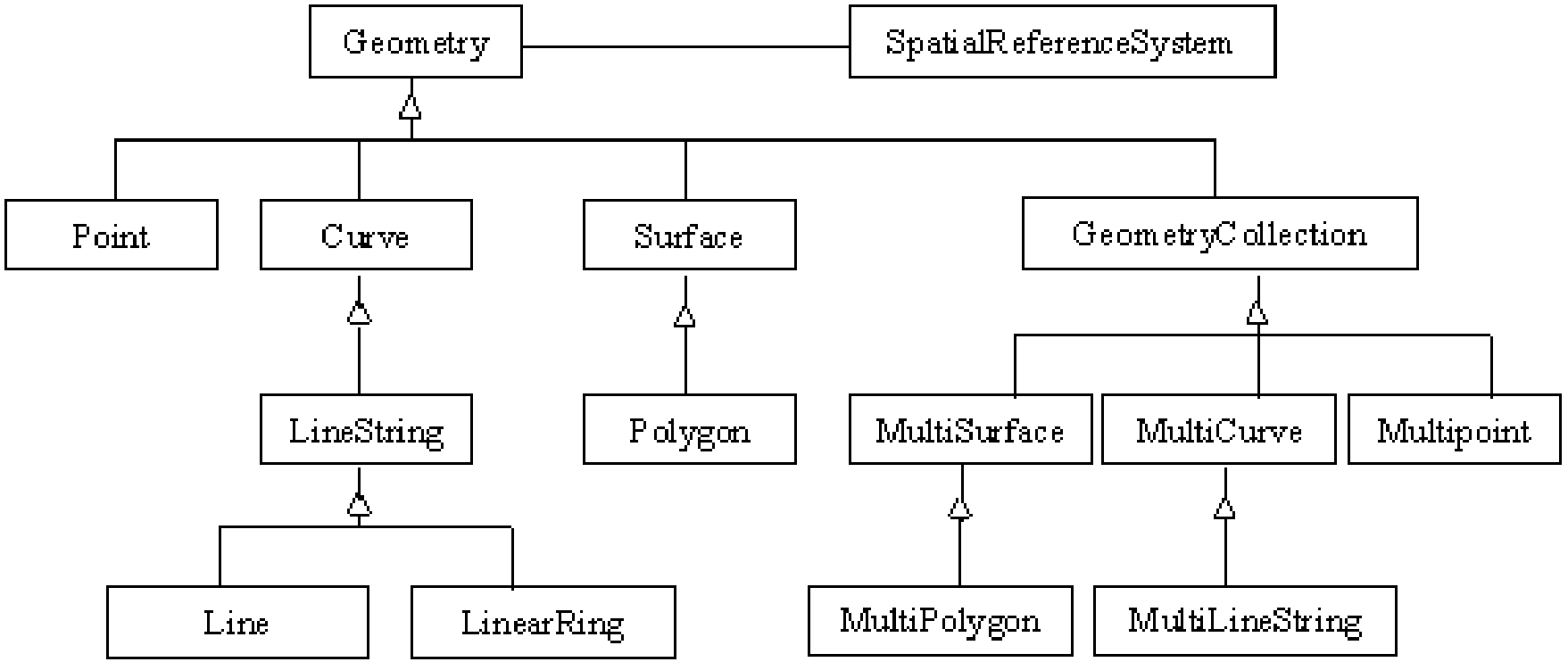
Norma ISO 19125

- OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture
  - Geometrický model (*Geometry Object Model*)
    - Nepodporuje rastrová data
  - Well-known Text Representation for Geometry
  - Well-known Binary Representation for Geometry
  - Well-known Text Representation of Spatial Reference Systems

Norma ISO 19125

- OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 2: SQL option
  - Architecture — SQL implementation using predefined data types
  - Architecture — SQL implementation using Geometry Types

Geometrický model OGC



Geometrický model OGC

- Objektový model pro geometrie tzv. ***Simple Features*** – prvky, jejichž geometrie se nesebeprotínají a u kterých je pro spojení dvou bodů geometrického popisu použita lineární interpolace
- Používá UML notaci
- Základní třídou je **Geometry** (abstraktní třída geometrických objektů); má čtyři podtřídy:
  - **Point** – nuladimenzionální geom. objekty
  - **Curve** – jednodimenzionální geom. objekty
  - **Surface** – dvoudimenzionální geom. objekty
  - **GeometryCollection** (multimnožina – např. kolekce bodů)

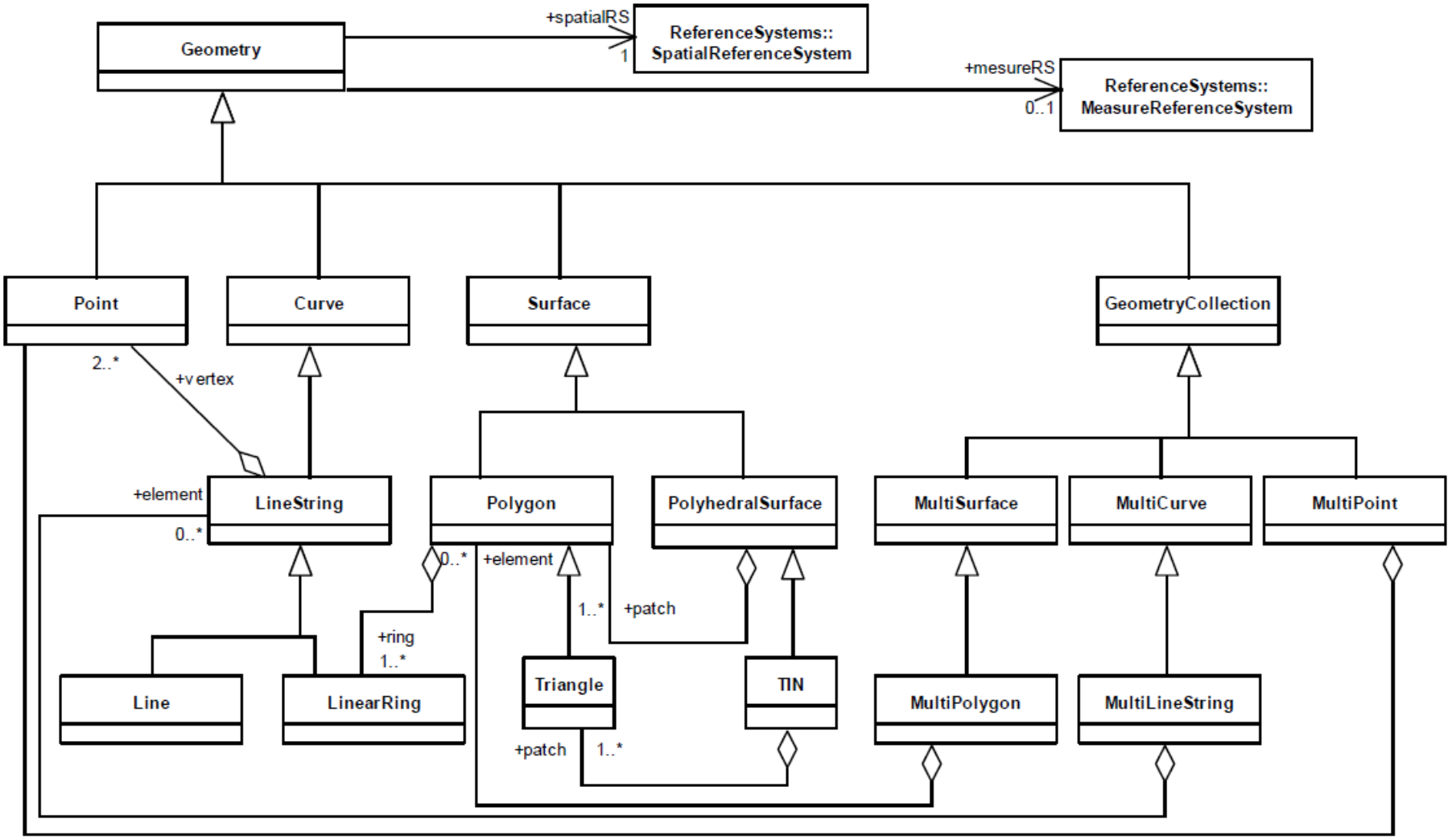
Geometrický model OGC

- Každý geometrický objekt je asociován s nějakým souřadnicovým systémem (v modelu mu odpovídá třída **Spatial Reference System**) – pro každý geometrický objekt musí být znám systém souřadnic.
- Křivka (**Curve**) je dána posloupností bodů, přičemž podtřídy (v příslušném OGC standardu definována pouze jedna) třídy **Curve** určují způsob interpolace
  - Podtřída **LineString** (*linie*, lomená čára) - lineární interpolace
- Objekty třídy **Line** (*úsečka*) jsou dány dvěma body.

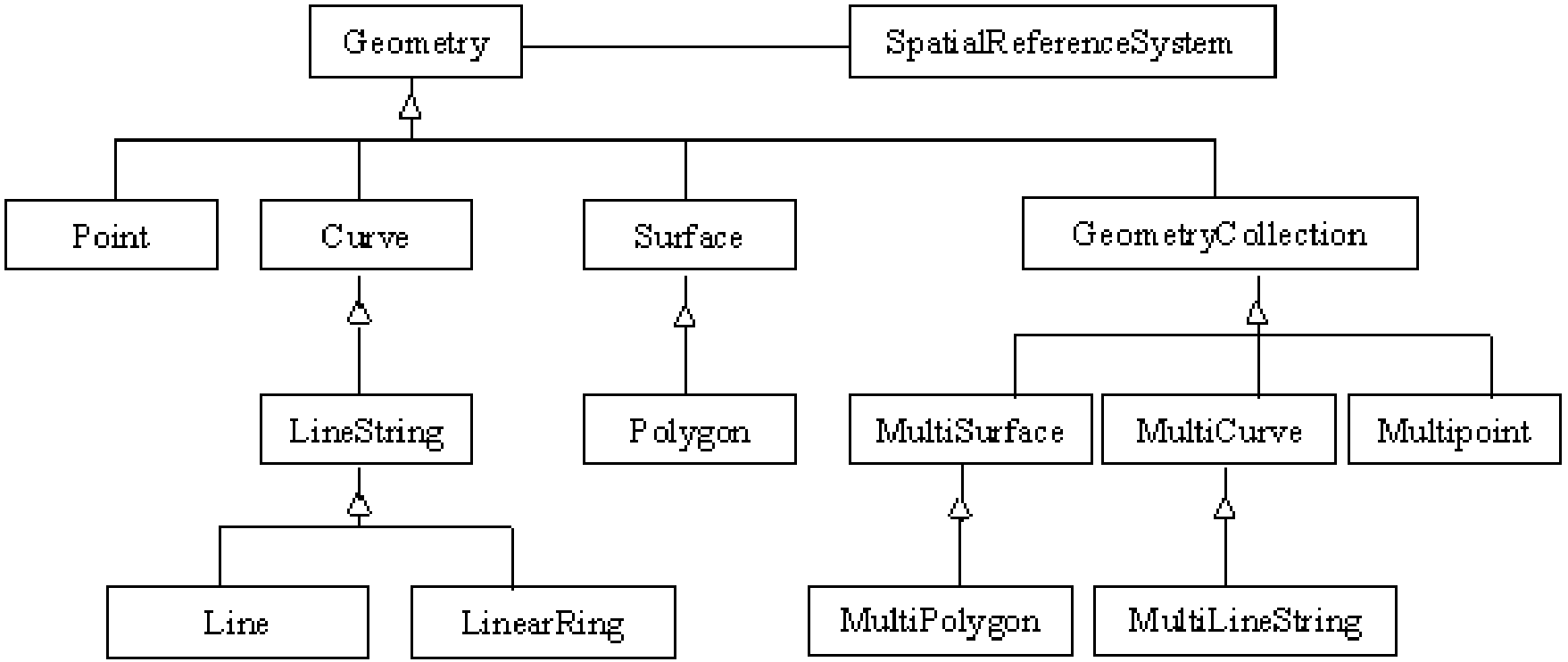
Geometrický model OGC

- Třída **LinearRing** obsahuje pouze jednoduché a uzavřené linie.
  - Křivka je jednoduchá, jestliže se sama se sebou nikde neprotíná
  - Uzavřené křivky mají společný počáteční a koncový bod
- Instance třídy **Polygon** jsou definovány vnější hranicí a několika vnitřními hranicemi (tvoří “díry”). Tyto hranice (*boundary*) jsou dány pomocí objektů ze třídy LinearRing.

Geometrický model OGC – včetně vztahů mezi třídami a integritními omezeními

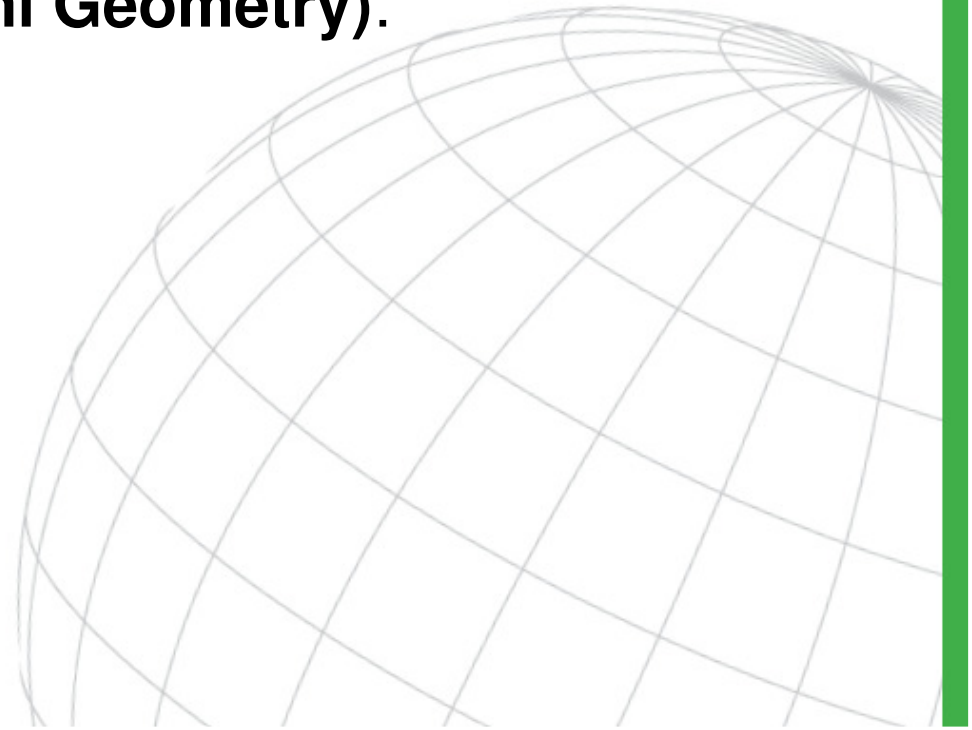


Geometrický model OGC



Geometrický model OGC – základní metody

- Pro jednotlivé třídy jsou specifikovány metody
  - **Základní**
  - **Pro porovnání dvou objektů**
  - **Podporující prostorové analýzy**
- Základní metody **(na úrovni Geometry)**:
  - Dimension();
  - GeometryType();
  - **SRID();**
  - **Envelope();**
  - **AsText();**



Geometrický model OGC – základní metody

- Základní metody **(na úrovni Geometry)**:
  - **AsBinary();**
  - IsEmpty();
  - IsSimple();
  - Is3D();
  - IsMeasured();
  - Boundary();

Geometrický model OGC – metody pro porovnání dvou objektů

- Návratovou hodnotou je TRUE/FALSE
- Metody pro testování prostorových vztahů mezi dvěma geometrickými objekty:
  - **Equals** (anotherGeometry : Geometry)
  - **Disjoint** (anotherGeometry : Geometry)
  - **Intersects** (anotherGeometry : Geometry)
  - **Touches** (anotherGeometry : Geometry)
  - **Crosses** (anotherGeometry : Geometry)
  - **Within** (anotherGeometry : Geometry)

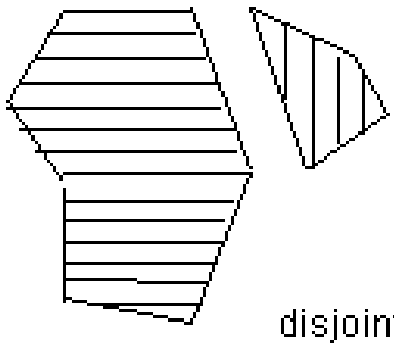
Geometrický model OGC – metody pro porovnání dvou objektů

- Metody pro testování prostorových vztahů mezi dvěma geometrickými objekty:
  - **Contains** (anotherGeometry : Geometry)
  - **Overlaps** (anotherGeometry : Geometry)
  - Relate (anotherGeometry: Geometry, intersectionPatternMatrix: String):
  - LocateAlong (mValue: Double)
  - LocateBetween (mValue: Double)

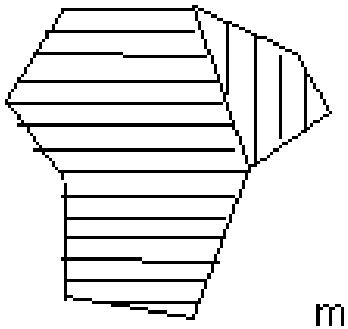
Binární prostorové predikáty

- Metody pro porovnání prostorových vztahů mezi dvěma geometrickými objekty založeny na **binárních predikátech**
- Vrací hodnotu `boolean` ⇒ binární prostorové predikáty tak mohou být přímo použity v klauzuli **`WHERE`**
- Jsou definovány pro všechny dimenze prostorových atributů
- Definice predikátů jsou částí modelu **DE-9IM** (*Dimensionally Extended Nine-Intersection Model*), který kombinuje možnosti průniků vnitřků, hranice a vnějšku dvou geometrických objektů

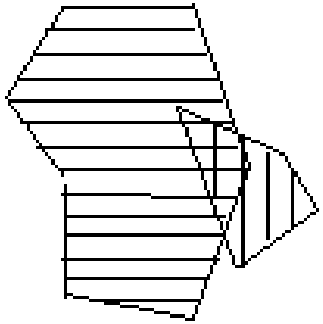
Binární prostorové predikáty



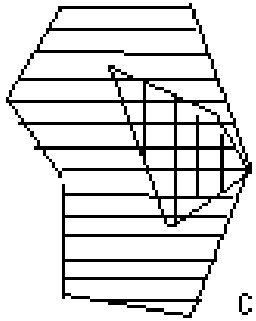
disjoint



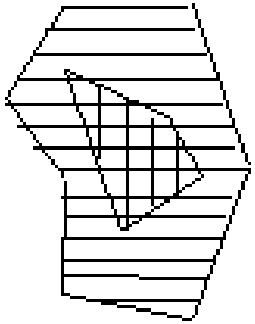
meet



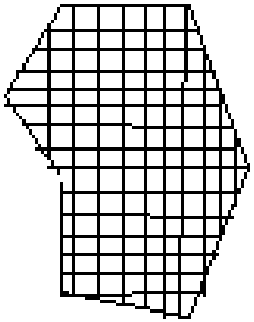
overlap



covers / coveredBy



inside / contains



equal

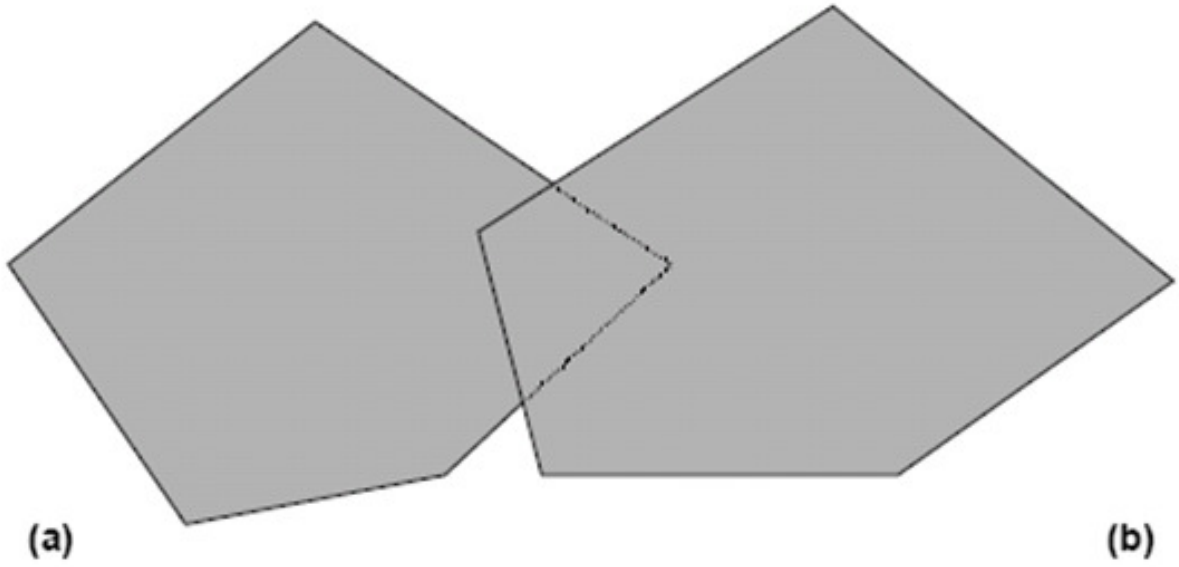
Model DE-9IM

	Interior	Boundary	Exterior
Interior	$dim(I(a) \cap I(b))$	$dim(I(a) \cap B(b))$	$dim(I(a) \cap E(b))$
Boundary	$dim(B(a) \cap I(b))$	$dim(B(a) \cap B(b))$	$dim(B(a) \cap E(b))$
Exterior	$dim(E(a) \cap I(b))$	$dim(E(a) \cap B(b))$	$dim(E(a) \cap E(b))$

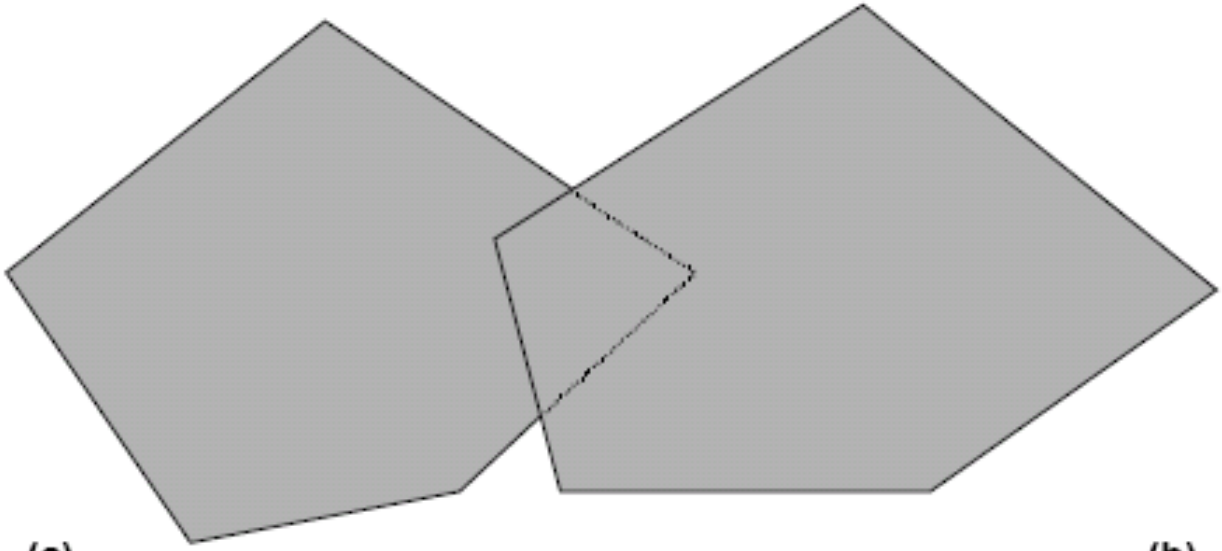
- Podstatné na tomto modelu je nejenom definice predikátů podle možných průniků, ale také **definice dimenze výsledku**

Cvičení

- Určete matici pro binární prostorový predikát INTERSECT



Binární prostorový predikát - INTERSECT



(a) (b)

	Interior	Boundary	Exterior
Interior	2	1	2
Boundary	1	0	1
Exterior	2	1	2

Geometrický model OGC – metody podporující prostorovou analýzu

- Na úrovni třídy **Geometry** jsou definovány tyto metody:
  - **Distance** (anotherGeometry: Geometry): Double
  - **Buffer** (distance: Double): Geometry
  - **ConvexHull** ( ): Geometry
  - **Intersection** (anotherGeometry: Geometry): Geometry
  - **Union** (anotherGeometry: Geometry): Geometry
  - Difference (anotherGeometry: Geometry): Geometry
  - SymDifference (anotherGeometry: Geometry): Geometry

Architektura – SQL implementace OGC modelu použitím předdefinovaných datových typů

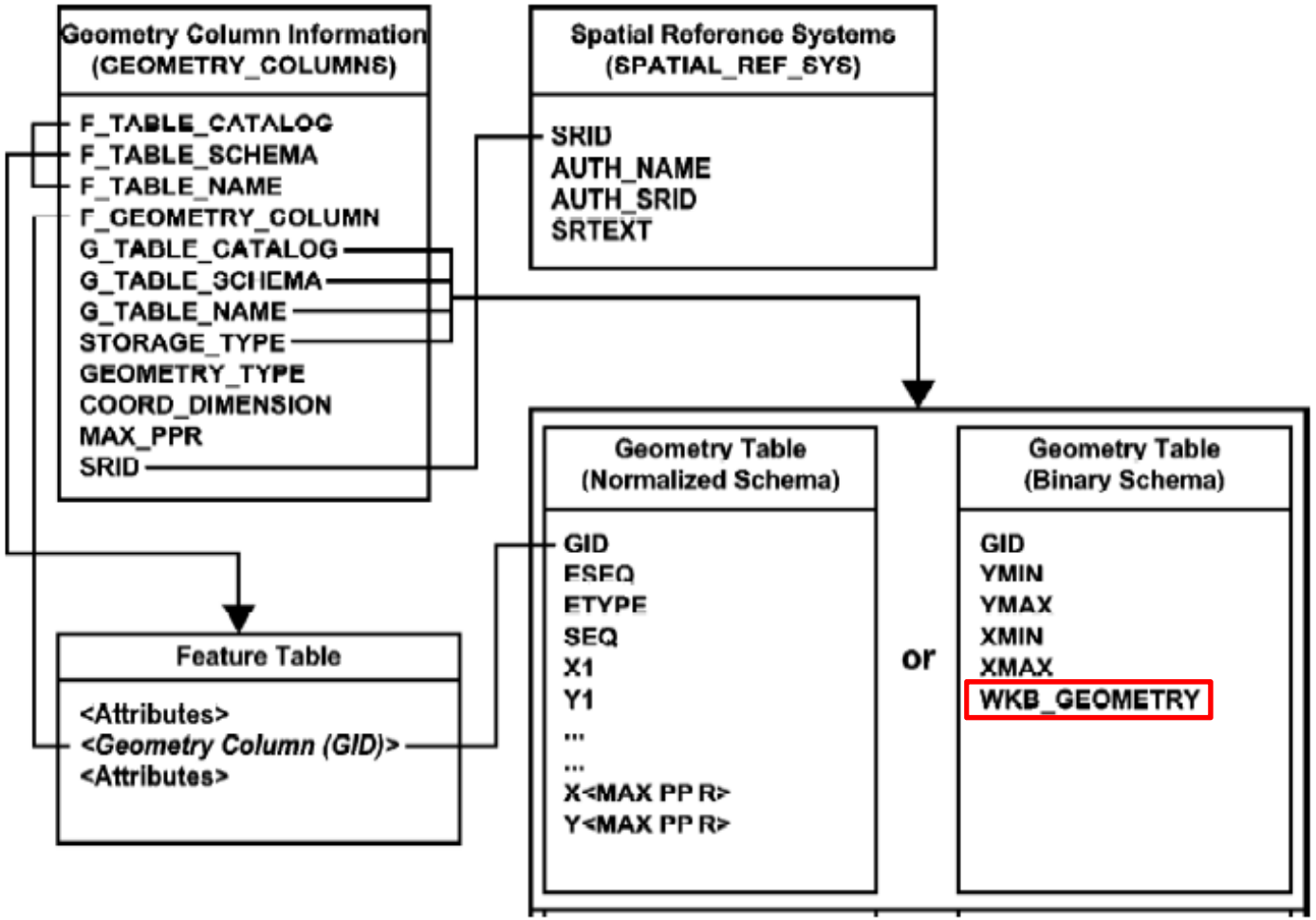
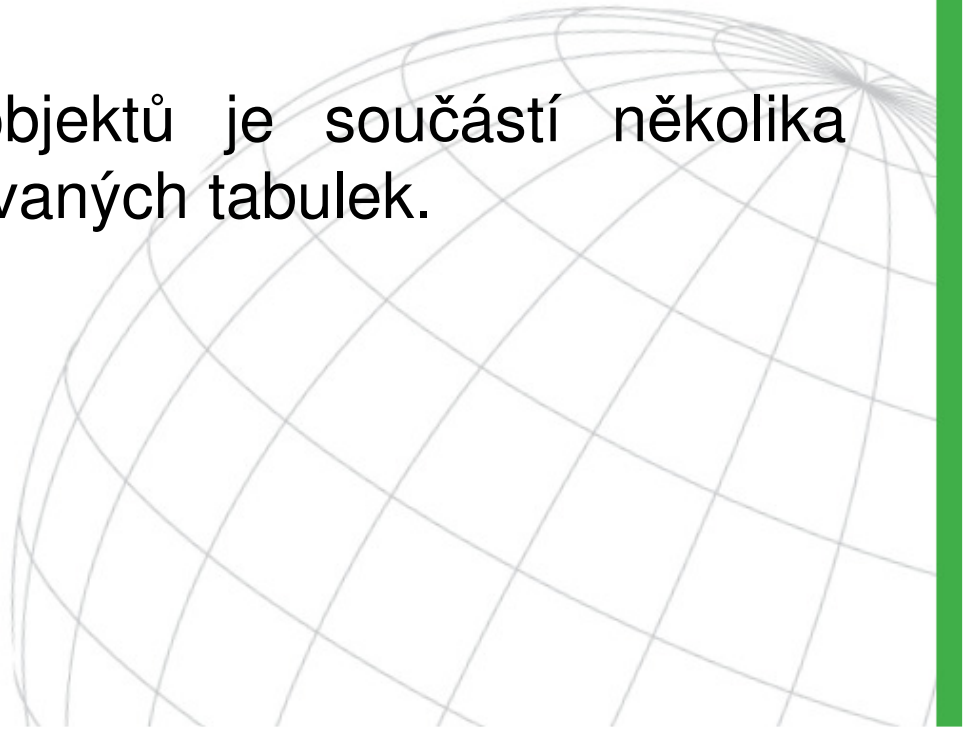
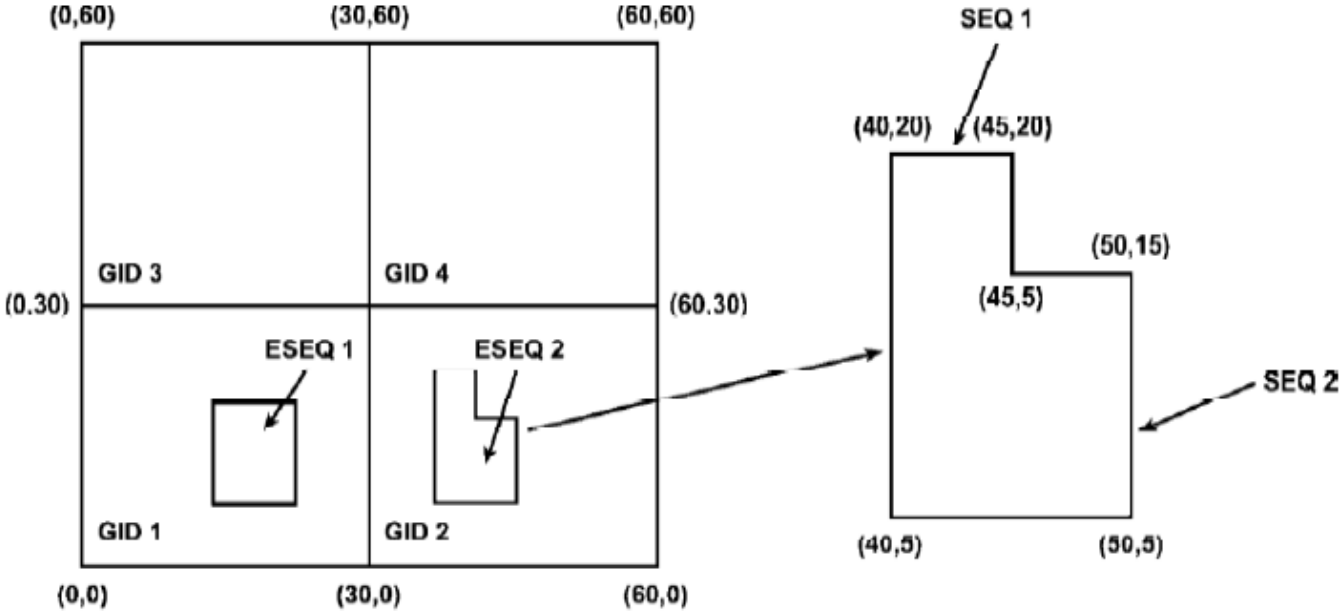


Figure 1: Schema for feature tables using predefined data types

Použití numerických typů pro uložení geometrických objektů

- Geometrické atributy jsou implementovány pomocí cizích klíčů do tabulky geometrických objektů.
- Hodnota geometrického atributu je implementována obecně pomocí sady několika řádků v tabulce geometrických objektů.
- Tabulka geometrických objektů je součástí několika dalších, klasicky normalizovaných tabulek.





GID 1	ESEQ	ETYPE	SEQ	X0	Y0	X1	Y1	X2	Y2	X3	Y3	X4	Y4
1	1	3	1	0	0	0	30	30	30	30	0	0	0
1	2	3	1	10	10	10	20	20	20	20	10	10	10
2	1	3	1	30	0	30	30	60	30	60	0	30	0
2	2	3	1	40	5	40	20	45	20	45	15	50	15
2	2	3	1	50	15	50	5	40	5	Nil	Nil	Nil	Nil
3	1	3	1	0	30	0	60	30	60	30	30	0	30
4	1	3	1	30	30	30	60	60	60	60	30	30	30

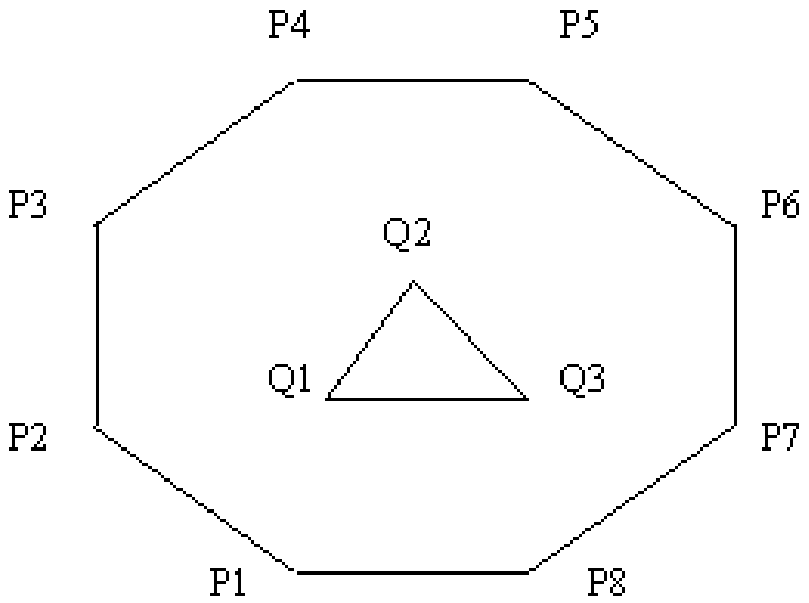
Figure 2: Example of geometry table for Polygon Geometry using SQL

Použití numerických typů pro uložení geometrických objektů

- Ukázka (Oracle 8, SQL92)
- Primitivní objekty z 2D jsou body, linie a polygony. *Elementy* těchto typů lze kombinovat do množin tvořících **geometrické objekty** identifikované **jednoznačným identifikátorem** (**GID**).
- Tyto objekty mohou mít i neprostorové atributy.
- Geometrické objekty se stejnými atributy se sdružují do *vrste*v. Vrstvy jsou uživatelsky orientované. Mohou být heterogenní, např. mosty (body) a silnice (linie), nebo homogenní (mosty a silnice jsou ve dvou vrstvách).

Použití numerických typů pro uložení geometrických objektů

- Namodelujme tento příklad (bez ADT,…):



Použití numerických typů pro uložení geometrických objektů

- Struktura relační databáze prostorového modulu se skládá ze čtyř relačních schémat:
  - **<jméno_vrstvy>_LAYER**(ORDCNT, LEVEL)
  - **<jméno_vrstvy>_DIM**(DIMNUM, LB, UB, TOLERANCE, DIMNAME)
  - **<jméno_vrstvy>_GEOM**(GID, ESEQ, TYPE, SEQ, X1, Y1, …, Xn, Yn)
  - **<jméno_vrstvy>_INDEX**(GID, CODE, MAXCODE)

Použití numerických typů pro uložení geometrických objektů

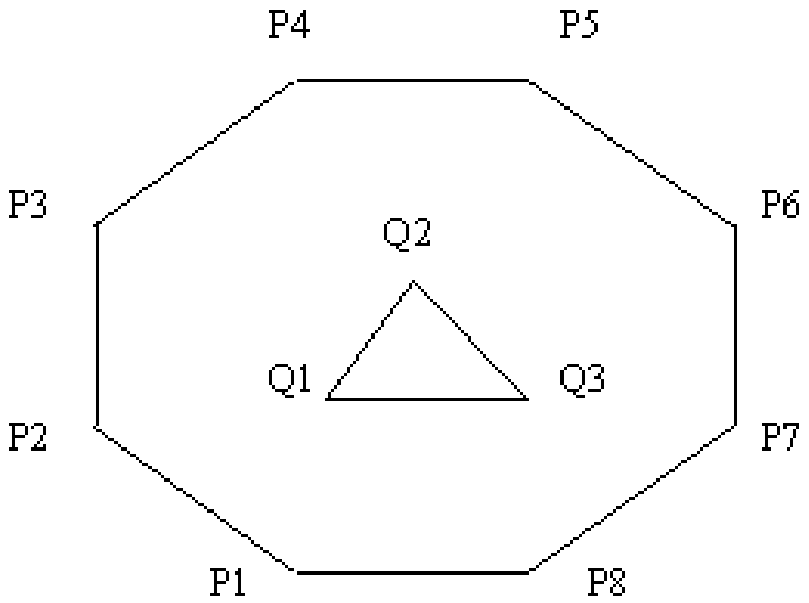
- Tabulka **<jméno_vrstvy>_LAYER**(ORDCNT, LEVEL)
  - Atribut ORDCNT udává počet souřadnic v každém řádku tabulky GEOM (geometrické objekty).
  - Ve sloupci LEVEL se uchovává informace o počtu dělení kvadrantů indexačních čtyřstromů.
- Tabulka **<jméno_vrstvy>_DIM**(DIMNUM, LB, UB, TOLERANCE, DIMNAME)
  - V tabulce DIM je popsána každá dimenze, tj. její číslo (DIMNUM), její minimální (LB) resp. maximální souřadnice (UB), tolerance (TOLERANCE), udávající přesnost rozlišení dvou bodů, a jméno dimenze (např. osa x, zeměpisná šířka apod.).

Použití numerických typů pro uložení geometrických objektů

- Tabulka **<jméno_vrstvy>_GEOM**(GID, ESEQ, TYPE, SEQ, X1, Y1, …, Xn, Yn)
  - V tabulce GEOM jsou v řádcích popsány části jednotlivých geometrických objektů.
  - Pro dané GID se udává číslo elementu ESEQ, typ elementu ETYPE (**0 pro bod, 1 pro úsečku, 2 pro polygon**), pořadí řádku SEQ v rámci jednoho elementu (např. úsečky polygonu se řadí buď ve směru nebo proti směru hodinových ručiček).
  - Zbytek sloupců tvoří souřadnice jednotlivých částí primitivního elementu. Např. pro osmiúhelník v tabulce GEOM je třeba 8 řádků a v každém řádku 4 souřadnice pro jednu z úseček, ze kterých se osmiúhelník skládá.
- Tabulka **<jméno_vrstvy>_INDEX**(GID, CODE, MAXCODE)

Použití numerických typů pro uložení geometrických objektů

- Jak vypadají jednotlivé tabulky pro následující objekt (polygon s dírou)?



Použití numerických typů pro uložení geometrických objektů

- Tabulky `LAYER` a `DIM`:

REGION1_LAYER
ORDCT
4

REGION1_DIM				
DIMNUM	LB	UB	TOLERANCE	DIMNAME
1	0	100	0,4	X osa
2	0	100	0,4	Y osa

Použití numerických typů pro uložení geometrických objektů

- Tabulka `GEOM`:

REGION1_GEOM							
GID	ESEQ	ETYPE	SEQ	X1	Y1	X2	Y2
1234	0	1	0	P1(X)	P1(Y)	P2(X)	P2(Y)
1234	0	1	1	P2(X)	P2(Y)	P3(X)	P3(Y)
1234	0	1	2	P3(X)	P3(Y)	P4(X)	P4(Y)
1234	0	1	3	P4(X)	P4(Y)	P5(X)	P5(Y)
1234	0	1	4	P5(X)	P5(Y)	P6(X)	P6(Y)
1234	0	1	5	P6(X)	P6(Y)	P7(X)	P7(Y)
1234	0	1	6	P7(X)	P7(Y)	P8(X)	P8(Y)
1234	0	1	7	P8(X)	P8(Y)	P2(X)	P2(Y)
1234	1	1	0	Q1(X)	Q1(Y)	Q2(X)	Q2(Y)
1234	1	1	1	Q2(X)	Q2(Y)	Q3(X)	Q3(Y)
1234	1	1	2	Q3(X)	Q3(Y)	Q1(X)	Q1(Y)

Použití numerických typů pro uložení geometrických objektů

- Dotazování se provádí pomocí těchto 4 tabulek a jejich interakce s oknem dotazu.
- Pro okno se musí opět vytvořit vrstva s odpovídajícími tabulkami. Dotaz je pak formulován v SQL nad těmito tabulky.
- Uvedené tabulky se podstatně změní v objektově relačním prostředí, jak je to realizováno v ORACLE 8i.
- Tam lze element reprezentovat jedním řádkem tabulky. Množina úseček osmiúhelníka se modeluje pomocí pole umístěného v jedno sloupci tabulky.

Použití binárních typů pro uložení geometrických objektů

Table 1: Example of geometry table for Polygon Geometry Using the Well-known Binary Representation for Geometry

GID	XMIN	YMIN	XMAX	YMAX	Geometry
1	0	0	30	30	< WKBGeometry >
2	30	0	60	30	< WKBGeometry >
3	0	30	30	60	< WKBGeometry >
4	30	30	60	60	< WKBGeometry >

Well-Known Binary Representation for Geometry

- Každý geometrický typ má svůj kód:

Type	Code
Geometry	0
Point	1
LineString	2
Polygon	3
MultiPoint	4
MultiLineString	5
MultiPolygon	6
GeometryCollection	7
CircularString	8
CompoundCurve	9
CurvePolygon	10
MultiCurve	11
MultiSurface	12
Curve	13
Surface	14
PolyhedralSurface	15
TIN	16

Použití binárních typů pro uložení geometrických objektů

- Základním stavebním blokem je reprezentace pro bod (**Point**); reprezentace ostatních geometrických typů jsou „vystavěny“ použitím již definovaných reprezentací geometrických objektů

```
// Basic Type definitions
// byte : 1 byte
// uint32 : 32 bit unsigned integer (4 bytes)
// double : double precision number (8 bytes)

// Building Blocks : Coordinate, LinearRing
Point {
  double x;
  double y}
```

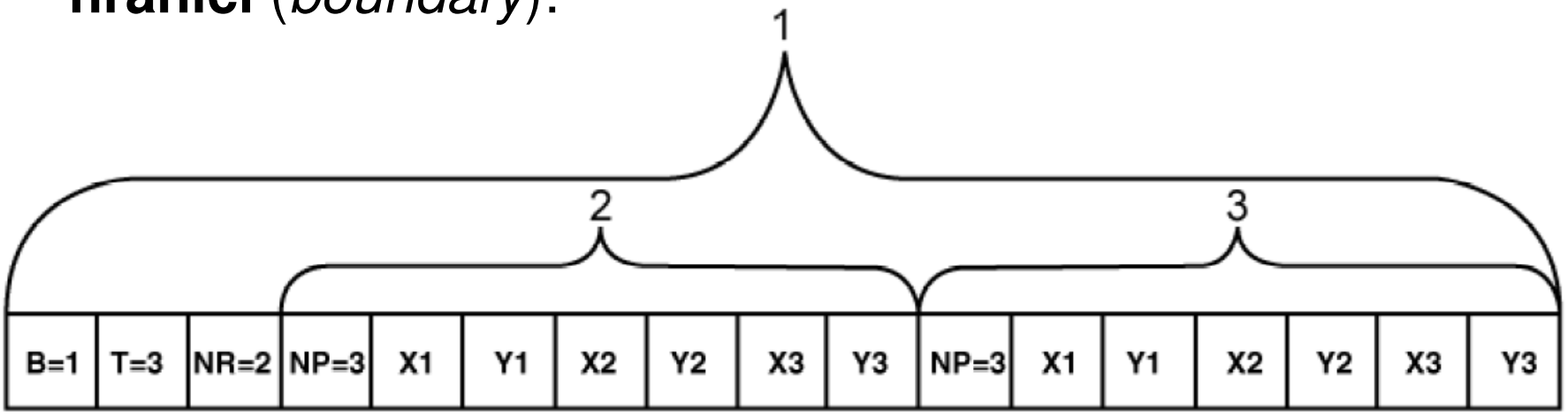
Použití binárních typů pro uložení geometrických objektů

```
WKBPoint {
  byte     byteOrder;
  static   uint32       wkbType = 1;
  Point    point}
```

```
WKBPolygon {
  byte                byteOrder;
  static uint32       wkbType = 3;
  uint32              numRings;
  LinearRing          rings[numRings]}
```

Použití binárních typů pro uložení geometrických objektů

- **Polygon** s jednou **vnější** (*outer*) a jednou **vnitřní** (*inner*) **hranicí** (*boundary*):



Key

1 WKB Polygon
2 ring 1
3 ring 2

Figure 25: Well-known Binary Representation for a geometric object in NDR format (B = 1) of type Polygon (T = 3) with 2 LinearRings (NR = 2) each LinearRing having 3 points (NP = 3)

Architektura — SQL implementace OGC modelu použitím prostorových datových typů (*geometry types*)

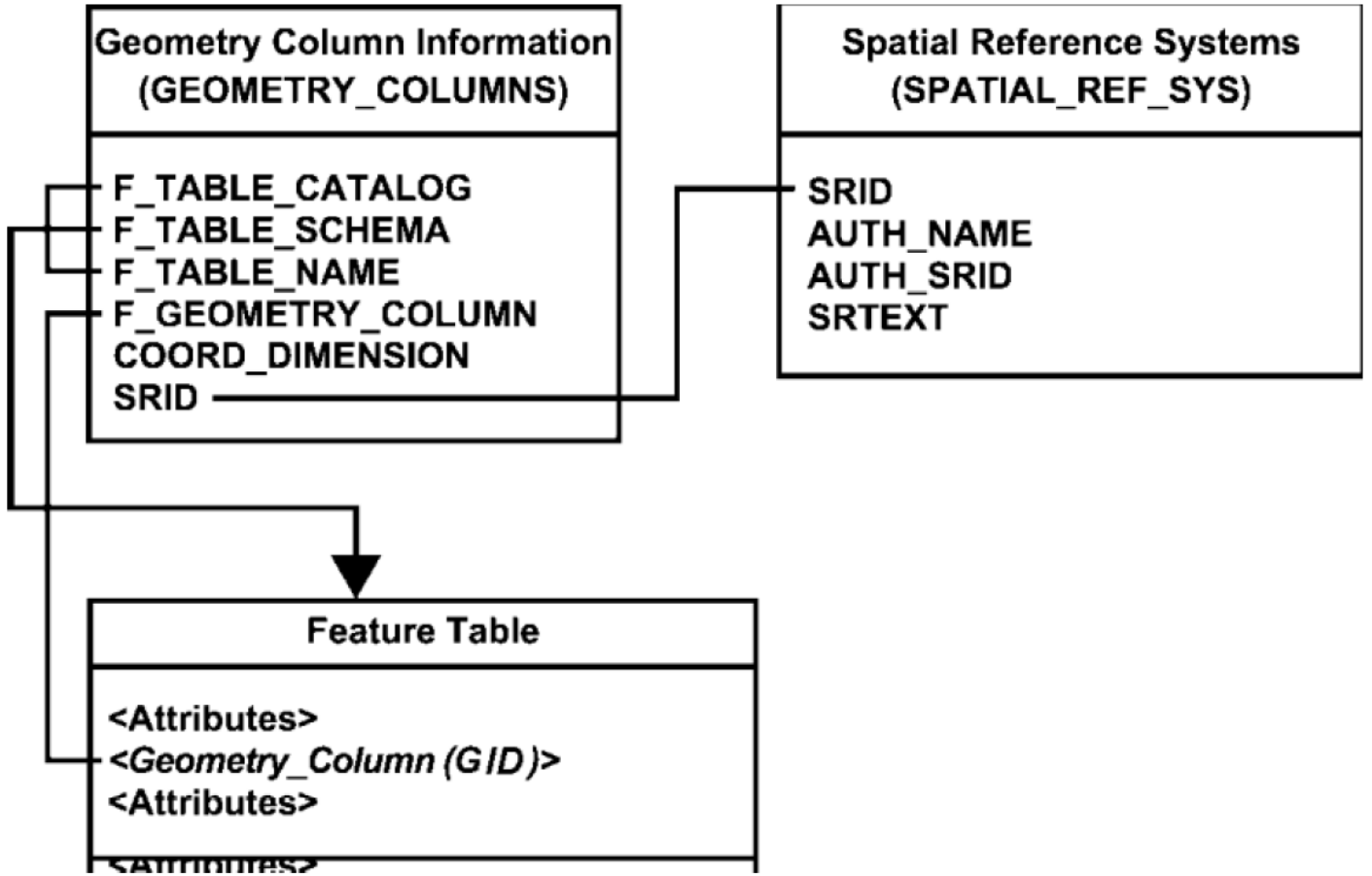


Figure 3: Schema for feature tables using SQL with Geometry Types

Použití textového přístupu ke geometriím objektu

- SQL92 s geometrickými typy je chápán jako SQL92 rozšířený o geometrické typy.
- V tomto případě je sloupec geometrických hodnot implementován přímo pomocí hodnot geometrického typu.
- Textová reprezentace znamená, že je možné zacházet z prostorovými daty přirozeně jako s literály. Např.

‘**LINESTRING (10 10, 20 20, 30 40)**‘

označuje linii danou třemi body.

Well-known Text Representation for Geometry

- Každý geometrický typ má textovou reprezentaci (*Well-known Text Representation*)

Geometry Type	Text Literal Representation	Comment
Point	Point (10 10)	a Point
LineString	LineString (10 10, 20 20, 30 40)	a LineString with 3 points
Polygon	Polygon ((10 10, 10 20, 20 20, 20 15, 10 10))	a Polygon with 1 exteriorRing and 0 interiorRings
Multipoint	MultiPoint ((10 10), (20 20))	a MultiPoint with 2 points

Použití ADT pro prostorové objekty

- Vzhledem k tomu, že již existuje potřebný standard SQL:1999, je vhodné vložit geometrické typy do SQL pomocí mechanismu ADT
- V objektově relačním SQL:1999 existují dva hlavní rysy objektového rozšíření relačního modelu dat:
  1. **ADT**
     - Je možné deklarovat nové typy sloupců včetně funkcí.
  2. **Typy řádků**
     - Uživatel může formulovat typy a funkce pro řádky tabulek.
     - Komponenty řádku mohou být základního typu, ADT, typu REF (odkazy), či typu, který modeluje vícehodnotové atributy (ARRAY).

Implementace ADT

```
CREATE TYPE t_zamestnanec AS (
   cislo_zam            INTEGER,
   jmeno                CHAR (20),
   adresa               t_adresa,
   vedouci              t_zamestnanec,
   datum_nastupu        DATE,
   zakladni_plat        DECIMAL(6,2)),
   bydliste             ST_GEOMETRY
   INSTANTIABLE NOT FINAL,
   METHOD odpracovana_leta( ) RETURNS INTEGER; /*jen signatury*/
   METHOD mzda ( )            RETURNS DECIMAL;
);
CREATE METHOD odpracovana_leta FOR typ_zamestnanec
BEGIN … END ;

CREATE METHOD mzda FOR typ_zamestnanec
BEGIN … END;
```

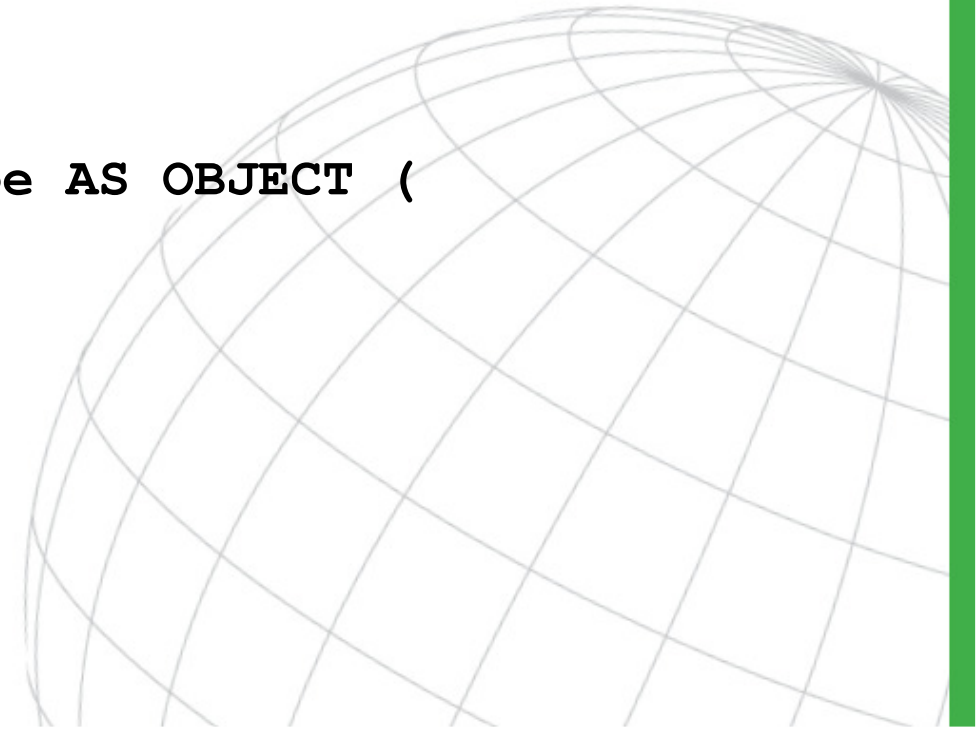
Implementace ADT - Oracle

- Syntax má tvar:

```
CREATE TYPE t AS OBJECT (
        seznam atributů a metod
);
/
```

- Př. ADT v Oracle:

```
CREATE TYPE PointType AS OBJECT (
        x       NUMBER,
        y       NUMBER,
);
/
```



Implementace ADT - Oracle

- Objekt může být použit jako ostatní typy v deklaracích objektových typů nebo tabulkových typů
- Př.

```
CREATE TYPE LineType AS OBJECT(
        end1            PointType,
        end2            PointType,
        MEMBER FUNCTION length(scale IN NUMBER)
                                RETURN NUMBER
);
/
CREATE TABLE Lines (
        lineID          INT,
        line            LineType
);
/
```

Implementace ADT - Oracle

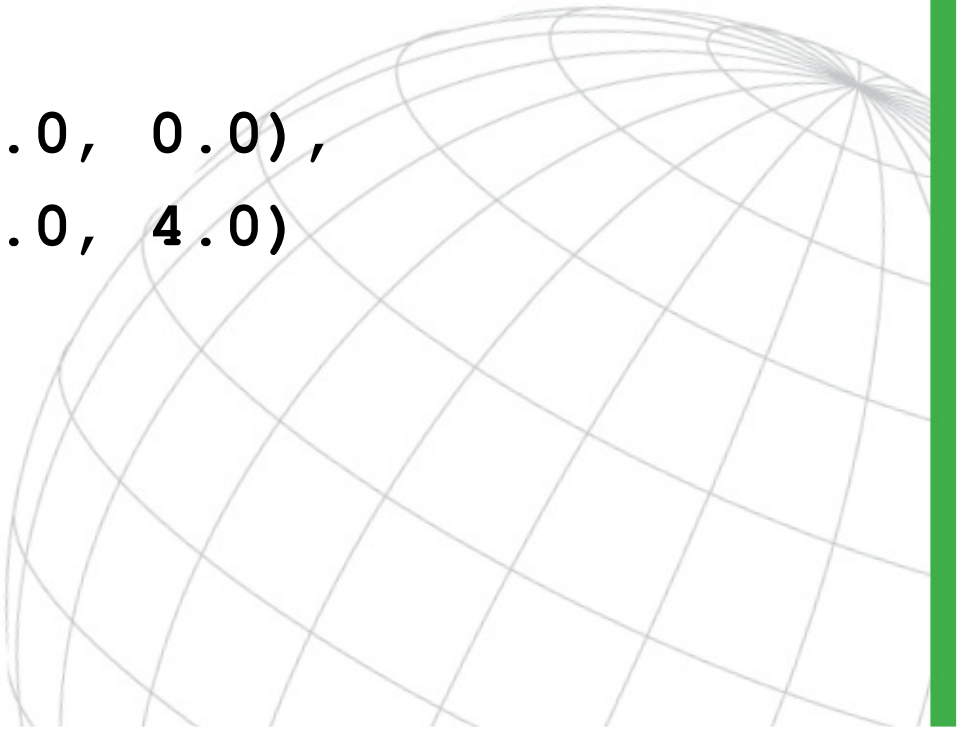
- Všechny metody pro typ jsou definovány v jednom `CREATE BODY` příkazu.
- Př.

```
CREATE TYPE BODY LineType AS
  MEMBER FUNCTION length(scale NUMBER) RETURN NUMBER IS
  BEGIN
     RETURN scale *
       SQRT((SELF.end1.x-SELF.end2.x) * (SELF.end1.x-SELF.end2.x)
             +
            (SELF.end1.y-SELF.end2.y) * (SELF.end1.y-SELF.end2.y)
           );
  END;
END;
/
```

Implementace ADT - Oracle

- Vytváření hodnot objektů - konstruktory:

```
INSERT INTO Lines VALUES
    (
      27, LineType
         (
           PointType(0.0, 0.0),
           PointType(3.0, 4.0)
         )
     );
```

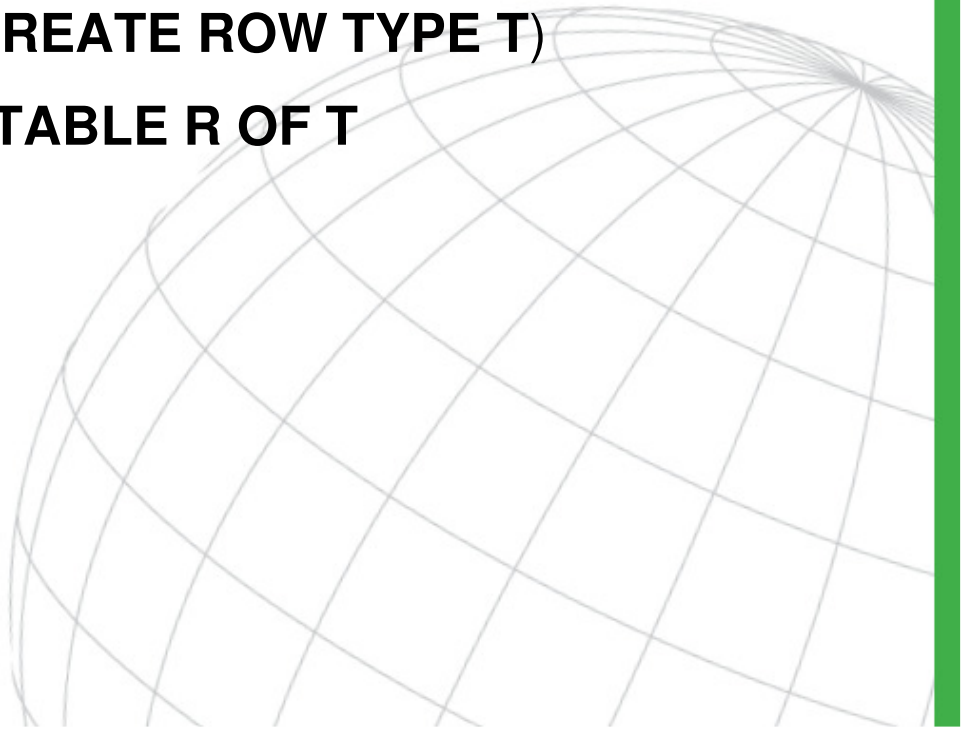


Použití *Typů řádků* pro prostorové objekty

- Tabulku lze definovat pomocí příkazu **CREATE TABLE** buď:
  1. přímo výčtem atributů,
  2. nebo se nejprve vytvoří:

     typ řádku (pomocí příkazu **CREATE ROW TYPE T**)

     a pak tabulka jako **CREATE TABLE R OF T**

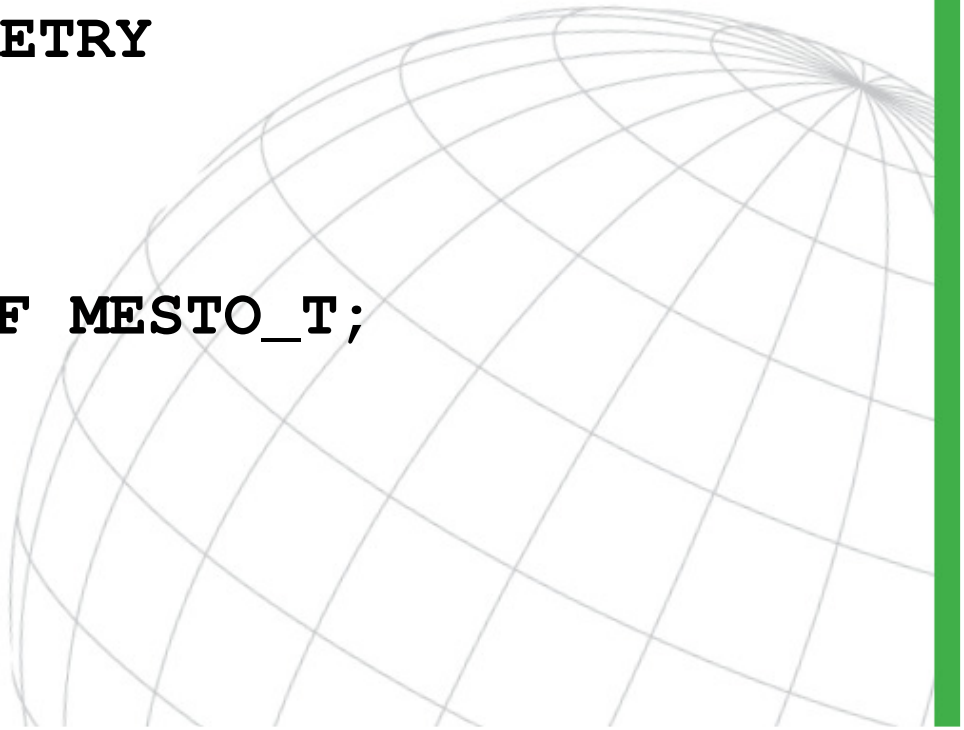


Definice tabulky pomocí *Typů řádků*

- **CREATE ROW TYPE MESTO_T (**

```
    jmeno       VARCHAR(30),
    populace    INTEGER,
    parky       VARCHAR(30) ARRAY[10],
    umisteni    ST_GEOMETRY
);
```

- **CREATE TABLE MESTA OF MESTO_T;**



Vložení dat do ADT – použití konstruktoru

- Vložení řádku do tabulky `MESTO` může být realizováno následujícím programem, přičemž předpokládáme, že atribut `UMISTENI` je typu typu `Point`.

```
BEGIN
DECLARE umisteni_mesta ST_POINT;
SET umisteni_mesta = NEW ST_POINT(100.0, 200.0);

INSERT INTO MĚSTA VALUES (
       'Plzeň',
       167500,
       ARRAY ['Sady svobody'],
       umisteni_mesta);
END;
```

Práce s ADT

- Dotazy na geometrické vlastnosti objektů umístěné v tabulkách se formulují pomocí běžných prostředků SQL spolu s využitím metod.
- Příklad: Najdi x-souřadnici města Plzně.

```
SELECT umisteni.X
FROM MESTA
WHERE JMENO = 'Plzeň';
```

Zdroje

- JANEČKA, K.: *Zajištění konzistence prostorových dat v Informačním systému katastru nemovitostí*. In: Proceedings of GIS Ostrava 2008. Tanger. Ostrava, 2008. s. 1-8. ISBN 978-80-254-1340-1.
- KOLINGEROVÁ, I.: *Přednášky k předmětu Vybrané algoritmické metody*. FAV ZČU v Plzni.
- MURRAY, Ch.: *Oracle Spatial Developer's Guide, 11g Release 1 (11.1).* Oracle. 2009.
- POKORNÝ, J.: *Prostorové datové struktury a jejich použití pro indexaci prostorových objektů*. In: Proceeding of GIS Ostrava 2000. Ostrava, 2000.
- POKORNÝ, J.: *Prostorové objekty a SQL*. In. Proceedings of GIS Ostrava 2001. Ostrava, 2001. ISSN: 1213-239X.
- POKORNÝ, J.; ŽEMLIČKA, M.: *Základy implementace souborů a databází*. Karolinum. Praha, 2004. 211 s. ISBN: 80-246-0837-5.
- ŽEMLIČKA, M.: *Přednášky k předmětu Organizace a zpracování dat II*. MFF UK v Praze.