Výuková prezentace 2

6MDBS1

Databázové systémy

Ing. Vladimír Přibyl, Ph.D.

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Další příkazy SQL

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Data Definition Language

- Obsahuje příkazy **CREATE, ALTER, DROP** pro práci s objekty databáze
  - TABULKA
  - VIEW
    - Pomocné výběrové dotazy
  - PROCEDURE
  - TRIGGER
  - INDEX
  - DATABASE
  - SEQUENCE

DDL – práce s tabulkami

- Pro definici nových tabulek slouží příkaz **CREATE TABLE**
- Pro změnu existující tabulky, tj. přidání, odebrání slupce či omezení slouží příkaz **ALTER TABLE**
- Pro odstranění tabulky slouží příkaz **DROP TABLE**
- Pro vyprázdnění tabulky slouží příkaz **TRUNCATE TABLE**

Definice nové tabulky

- Základní formát příkazu (hodnoty v [] nejsou povinné)

```
CREATE TABLE NazevTabulky (
        NazevSloupce [DatovyTyp] [Omezeni sloupce] [VychoziHodnota],
        NazevSloupce [DatovyTyp] [Omezeni sloupce] [VychoziHodnota],
        NazevSloupce [DatovyTyp] [Omezeni sloupce] [VychoziHodnota],
        …
        [Omezení tabulky]
        …

);
```

Typy omezení sloupců a tabulek

- NOT NULL (jen u sloupců)
  - hodnota musí být zadána
- NULL (jen u sloupců, default možnost, není potřeba ji uvádět)
  - hodnota nemusí být zadána
- IDENTITY(m,n)
  - definice náhradního klíče, m - počáteční hodnota, n - krok
- UNIQUE
  - jsou požadovány jedinečné hodnoty
- PRIMARY KEY
- FOREIGN KEY
- CHECK

Definice nové tabulky

- Příklad (bez omezení tabulky):

```
CREATE TABLE Osoba (
        ID              Integer         PRIMARY KEY,
        Jmeno           VarChar(50)     NOT NULL,
        Prijmeni        VarChar(50)     NOT NULL,
        DatumNarozeni   Date            NOT NULL,
        PočetRolí       Date            NOT NULL DEFAULT 0,

        …
);
```

Definice omezení tabulky

- Pokud použijeme omezení u sloupce, týká se jen tohoto sloupce. Omezení tabulky je definováno samostatně a může se týkat i více sloupců zároveň
- Lze jej také samostatně z definice tabulky odstranit, používá se často i pro jednotlivé slupce
- Syntaxe:

CONSTRAINT NazevOmezeni [klauzule spojené s příslušným omezením tabulky]

Definice nové tabulky

- Příklad s omezením tabulky, ID je náhradním klíčem:

```
CREATE TABLE Osoba (
        ID                  Integer         IDENTITY(1,1),
        Jmeno               VarChar(50)     NOT NULL,
        Prijmeni            VarChar(50)     NOT NULL,
        DatumNarozeni       Date            NOT NULL,
        PočetRolí           Date            NOT NULL DEFAULT 0,

        CONSTRAINT Osoba_PK PRIMARY KEY (ID)



);
```

Definice omezení FOREIGN KEY

- Jedná se o způsob realizace omezení referenční integrity (bez TRIGGERu).
- Může být použito jak u sloupce (jednoduchý klíč), tak u tabulky (může být i složený).
- Syntaxe umožňuje definovat odkaz na referenční sloupec jiné tabulky a také akci, která může nastat, pokud dojde ke změně nebo odstranění odkazované hodnoty.
- Obecná syntaxe:

CONSTRAINT <NazevOmezeni>
FOREIGN KEY (<sloupec> [{,<sloupec>} …])
REFERNCES <odkazovaná tabulka> [(<odkazované sloupce>)]
[MATCH {FULL|PARTIAL|SIMPLE}]
[<spouštěná referenční akce>]

Klauzule MATCH

- Má smysl jen v případě složeného cizího klíče.
- Souvisí s možností hodnot NULL.
  - MATCH FULL – buď musí být všechny sloupce NULL nebo žádný
  - MATCH PARTIAL – 1 nebo více sloupců může mít hodnotu NULL, ostatní musí mít hodnotu z odkazovaných sloupců.
  - MATCH SIMPLE – pokud má alespoň jeden sloupec hodnotu NULL, pak ostatní mohou mít jakoukoli hodnotu.
    - tato možnost je default, pokud není klauzule v případě složeného cizího klíče použita.

<spouštěná referenční akce>

- Umožňuje definovat akce, které se provedou, pokud dojde k pokusu o aktualizaci, či odstranění odkazovaných dat.
- Syntaxe:

ON UPDATE <referenční akce> [**ON DELETE** <referenční akce>] |
ON DELETE <referenční akce> [**ON UPDATE** <referenční akce>]

<referenční akce>::= {CASCADE|SET NULL|SET DEFAULT|RESTRICT|NO ACTION}

Typy referenčních akcí

- CASCADE – mazání či aktualizace svázaných polí v kaskádě.
- SET NULL – po změně či odstranění odkazovaného pole se hodnota odkazujících se polí nastaví na hodnotu NULL.
- SET DEFAULT – po změně či odstranění odkazovaného pole se hodnota odkazujících se polí nastaví na nastavenou výchozí hodnotu.
- RESTRICT – mazání či aktualizace nejsou umožněny
- NO ACTION – mazání či aktualizace nejsou globálně umožněny, ale v průběhu provádění SQL příkazu k tomu může dočasně dojít. (default možnost)

Příklad definice FOREIGN KEY

```
CREATE TABLE Osoba (
        ID                  Integer         IDENTITY(1,1),
        Jmeno               VarChar(50)     NOT NULL,
        Prijmeni            VarChar(50)     NOT NULL,
        DatumNarozeni       Date            NOT NULL,
        PočetRolí           Date            NOT NULL DEFAULT 0,
        TeamID              Integer         NULL

        CONSTRAINT          Osoba_PK        PRIMARY KEY (ID),
        CONSTRAINT          Osoba_FK        FOREIGN KEY (TeamID)
                REFERENCES Týmy(TeamID) MATCH FULL
                ON UPDATE CASCADE ON DELETE NO ACTION


);
```

Definice omezení CHECK

- Může být použito pro sloupec i pro tabulku.
- Umožňuje zadat podmínku, která musí být splněna při zadávání hodnot.
- Pokud je použito pro sloupec, pak se podmínky musí týkat hodnoty zadávané do sloupce.
- Pokud je použito pro tabulku, pak se podmínky mohou týkat všech hodnot zadávaných do sloupců tabulky.
- Syntaxe v případě omezení tabulky:

CONSTRAINT NazevOmezeni CHECK (<vyhledávací podmínka>)

Definice omezení CHECK

- Jako vyhledávací podmínku je možné použít vše, co je možné použít v podmínce příkazu SELECT. (Viz dále) Může to tedy být jen jednoduché porovnání, ale také složitý výraz využívající predikáty a logické operátory.
- Příklad:

```
CREATE TABLE Osoba (
   ID                Integer          IDENTITY(1,1),
   Jmeno             VarChar(50)      NOT NULL,
   Prijmeni          VarChar(50)      NOT NULL,
   PočetRolí         Date             NOT NULL DEFAULT 0 CHECK (PočetRolí < 10),
   PočetZkoušek      Integer,

     CONSTRAINT      Osoba_PK         PRIMARY KEY (ID),
     CONSTRAINT      Osoba_CK         CHECK (PočetZkoušek >=1 AND PočetZkoušek <=5)
);
```

Změna existující tabulky

- Základní formát příkazu

```
ALTER TABLE NazevTabulky
        ADD COLUMN NazevSloupce  DatovyTyp  VolitelneOmezeni
        | ALTER COLUMN NazevSloupce
        { SET DEFAULT VychoziHodnota | DROP DEFAULT}
        | DROP COLUMN NazevSloupce {CASCADE | RESTRICT}
;
```

Odstranění existující tabulky

```
DROP TABLE NazevTabulky [{CASCADE | RESTRICT}];
```

Vyprázdnění existující tabulky

```
TRUNCATE TABLE NazevTabulky;
```

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

View

- Objekt umožňující specifikovaný pohled na data v jedné, nebo více tabulkách.
- Při každém odkazu na View se provádí příslušný příkaz SELECT pro aktuální data.
- View lze za určitých podmínek využít i pro aktualizaci záznamů.
- Vytváří se příkazem

CREATE VIEW <nazev> [({<nazvy sloupcu pohledu>})]
AS <příkaz SELECT>
[WITH CHECK OPTION]

- Názvy sloupců pohledu je potřeba uvést pokud:
  - některé hodnoty sloupců jsou založeny na výrazech, ne přímo na hodnotách sloupců z dotazu
  - existují duplicitní názvy (při propojování více tabulek)
- pozn. – pokud jsou názvy uvedeny, musí být uvedeny pro všechny sloupce

View

- Podmínky aktualizovatelnosti se vztahují k příkazu SELECT (není to explicitně nastavitelná vlastnost View)
- Zjednodušeně
  - SELECT nesmí obsahovat souhrny, agregační funkce
  - data nesmí být automaticky eliminována (DISTINCT)
  - každý sloupec i řádek musí být jednoznačně identifikovatelný v původních tabulkách
- S aktualizovatelností View souvisí i nepovinná klauzule **WITH CHECK OPTION**. Pokud příkaz SELECT obsahuje klauzuli WHERE a aktualizací by došlo k tomu, že aktualizované řádky již podmínku nesplní, přítomnost klauzule WITH CHECK OPTION aktualizaci zabrání

View

- View jsou uloženy v databázi trvale.
- Odstranění se provádí příkazem DROP VIEW <nazev>
- Změnu existujících View (příkaz ALTER VIEW) podporují jen některé databázové systémy (např. SQL server). Obecně v SQL není možná.

Rutiny spouštěné v SQL

- jsou to procedury nebo funkce spouštěné v SQL
- umožňují zadávat vstupní i výstupní parametry
- jsou to sady příkazů SQL jazyka
- způsob implementace se u jednotlivých databázových systémů mírně odlišuje od standardu SQL
- funkce se od procedury odlišuje především tím, že má návratovou hodnotu

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

SQL procedura

- Procedura se vytváří příkazem **CREATE PROCEDURE**
- Může obsahovat všechny prvky jazyka SQL včetně řídících příkazů a příkazů specifických pro daný SŘBD.
- Výsledkem procedury může být samozřejmě také množina záznamů.
- základní syntaxe:

CREATE PROCEDURE <nazev> ([<parametr> [{, parametr}…]])

[<popis procedury> …]

<tělo procedury>

SQL procedura

- parametry procedury mohou být obecně
  - vstupní (IN)
  - výstupní (OUT)
  - vstupně-výstupní (INOUT)
- Součástí definice parametru je i definice datového typu
- Syntaxe:

```
CREATE PROCEDURE ZmenitJmenoOsoby  (IN ID LONGINT, IN NoveJmeno VARCHAR,
OUT Uspesne BOOLEAN)
BEGIN
        UPDATE Osoby SET Jmeno = NoveJmeno WHERE IDOsoby = ID;
        …
        Uspesne = True;
END;
```

SQL procedura

- Volání (spouštění) procedury se provádí příkazem CALL

  CALL ZmenitJmenoOsoby (10,“Jana“, UspesneZmeneno)

- V rámci procedur je možné použít deklaraci lokálních proměnných (DECLARE) a také řídící příkazy
  - BEGIN – END (složené příkazy)
  - IF (podmínky)
  - LOOP, WHILE (cykly)
- Procedury je samozřejmě možné volat i v těle jiné procedury či funkce

SQL Funkce

- Vytváří se příkazem CREATE FUNCTION
- Syntaxe:

CREATE FUNCTION <nazev> ([<parametr> [{, parametr}…]])
[<popis funkce> …]
RETURNS <datový typ>

<tělo funkce>

- <tělo funkce> musí obsahovat klauzuli RETURN, která definuje výstupní hodnotu funkce
- Funkce se na rozdíl od procedury volá přímo ve výrazu zápisem názvu.

Modelování databází

Vývoj a modelování databází

- Součást komplexnějšího procesu vývoje informačního systému, pokud samozřejmě tento využívá práci s daty prostřednictvím databázového systému.
- Většina součástí podnikových informačních systémů je založena na zpracování dat.
- Zařazení návrhu databáze do procesu vývoje IS záleží na metodice tohoto procesu a celkovému přístupu k vývoji
  - Sekvenční přístup x agilní přístupy
  - Objektový návrh x neobjektový, strukturovaný návrh
- Často je návrh databáze poměrně samostatnou fází vývoje, založenou na neobjektovém přístupu.

Fáze návrhu databáze

- Konceptuální návrh
  - Vytvoření modelu dat, který odpovídá požadavkům
  - Nezávislý na způsobu realizace a volbě DBMS
  - Výsledkem je ER model
- Logický návrh
  - Převedení konceptuálního návrhu (ER modelu) do struktury odpovídající typu DBMS. Tedy v případě relačního DBMS do struktury relací (tedy tabulek) a vztahů mezi nimi.
  - Kontrola konzistence, integrity a redundance prostřednictvím normalizace datového modelu
- Fyzický návrh
  - Realizace relačního schématu vytvořeného při logickém návrhu v konkrétním prostředí DBMS
  - Realizace opatření k zajištění integrity a bezpečnosti dat, optimalizace výkonu, případná částečná denormalizace.

ER model

- Entity – Relationship model (model/diagram entit a vztahů)
- Základní komponenty
  - entity
  - atributy
  - vztahy
- Při návrhu se uplatňuje tzv. Top-down princip
  - nejprve identifikace entit
  - poté analýza vlastností entit – atributy
  - analýza a modelování vztahů

  - Obvykle se jedná o iterativní proces.

Entita

- Množina objektů se shodnými vlastnostmi
- Každý jednoznačně identifikovatelný objekt z této množiny se nazývá výskyt entity.
- Entity mohou být
  - Fyzické – odpovídají reálným objektům
  - Abstraktní – existují jen jako pojmy
- Entity můžeme rozdělit také na (podrobněji viz dále)
  - Silné (regulární, kmenové, základní) entity – jejich výskyty můžeme rozlišit bez potřeby dalších entit
  - Slabé (popisné) entity – jsou existenčně nebo identifikačně závislé na jiné/jiných entitách.

Atribut

- Vlastnost entity, případně vztahu mezi entitami
- Typy atributů
  - Jednoduchý atribut – skládá se jen z jedné komponenty
  - Složený atribut – skládá se z více komponent a lze jej rozložit na více jednoduchých atributů

  - Atribut s jednou hodnotou – jedna hodnota na jeden výskyt entity
  - Atribut s více hodnotami – více hodnot na jeden výskyt entity

  - Základní – hodnotu nelze odvodit z jiných atributů
  - Odvoditelný – hodnotu nelze odvodit z jiných atributů

  - Povinný (totální) – musí obsahovat hodnotu
  - Nepovinný (parciální) – nemusí obsahovat hodnotu

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Klíče

- Atributy nebo jejich kombinace, které slouží k identifikaci záznamů.
- Klíče mohou být jedinečné – identifikují jen jeden záznam, nebo nejedinečné – identifikují více záznamů.
- Klíč, který je složen z více atributů se nazývá složený klíč (může samozřejmě být jedinečný i nejedinečný).
- Jedinečný klíč, který neobsahuje nadbytečné atributy (nejsou potřeba pro identifikaci) se stává kandidátním klíčem. Kandidátních klíčů může být v relaci více. Pro identifikaci je pak vybrán jeden z nich a ten se tak stává primárním klíčem.
- Často se stává, že z atributů, které jsou v relaci a popisují reálné entity, není možné sestavit nebo vybrat jednoduchý kandidátní klíč. Proto je do relace často přidáván další umělý atribut, jehož unikátní hodnoty udržuje sám systém řízení databáze. Tento klíč se nazývá náhradní a používá se jako primární klíč.
- Primární klíč se často využívá i pro základní řazení záznamů relace (řádků tabulky).
- Cizí klíč je klíč v jedné tabulce, který odpovídá kandidátnímu (primárnímu) klíči v jiné (případně téže) tabulce.

Vztahy mezi entitami

- Vztah = množina smysluplných spojení mezi výskyty entit
- Stupeň vztahu určuje počet entit, které do vztahu vstupují
  - Unární vztah – vztah entity se sebou samou. Rekurzivní vztah, využívá se pro realizaci hierarchií.
  - Binární vztah – vztah mezi dvěma entitami (nejčastější možnost).
  - Ternární … vícenásobný (n-ární) vztah – tři a více entit
- Důležitou vlastností vztahu je jeho multiplicita.
  - Počet výskytů jedné entity, které se mohou vztahovat k jedinému výskytu související entity.

Multiplicita vztahu

- Multiplicita se skládá ze dvou samostatných částí
  - Kardinalita – vlastní počet výskytů entit, které do vztahu vstupují
    - 1 : 1
    - 1 : N
    - M : N
  - Participace – vyjadřuje fakt, zda do vztahu vstupují všechny výskyty entity, nebo ne
    - Totální (úplný) vztah
    - Parciální (neúplný) vztah

    - Participace je „směrová“ vlastnost, to znamená, že vztah mezi entitami může být v jednom směru totální a ve druhém parciální.

Funkční závislost

- Významová (sémantická) závislost mezi atributy nebo jejich skupinami v rámci jedné entity
- Atribut A je funkčně závislý na atributu B pokud
  - hodnota atributu B určuje hodnotu atributu A a
  - z hodnoty B zjistíme právě jednu hodnotu A.
- Značíme B→A
- Pojem funkční závislosti se využívá při normalizaci datového modelu

A,B,C → D hodnota D závisí na A,B,C

A → B,C,D hodnota B,C,D závisí na A

A,B → C,D hodnota C,D závisí na A,B

Slabé entity

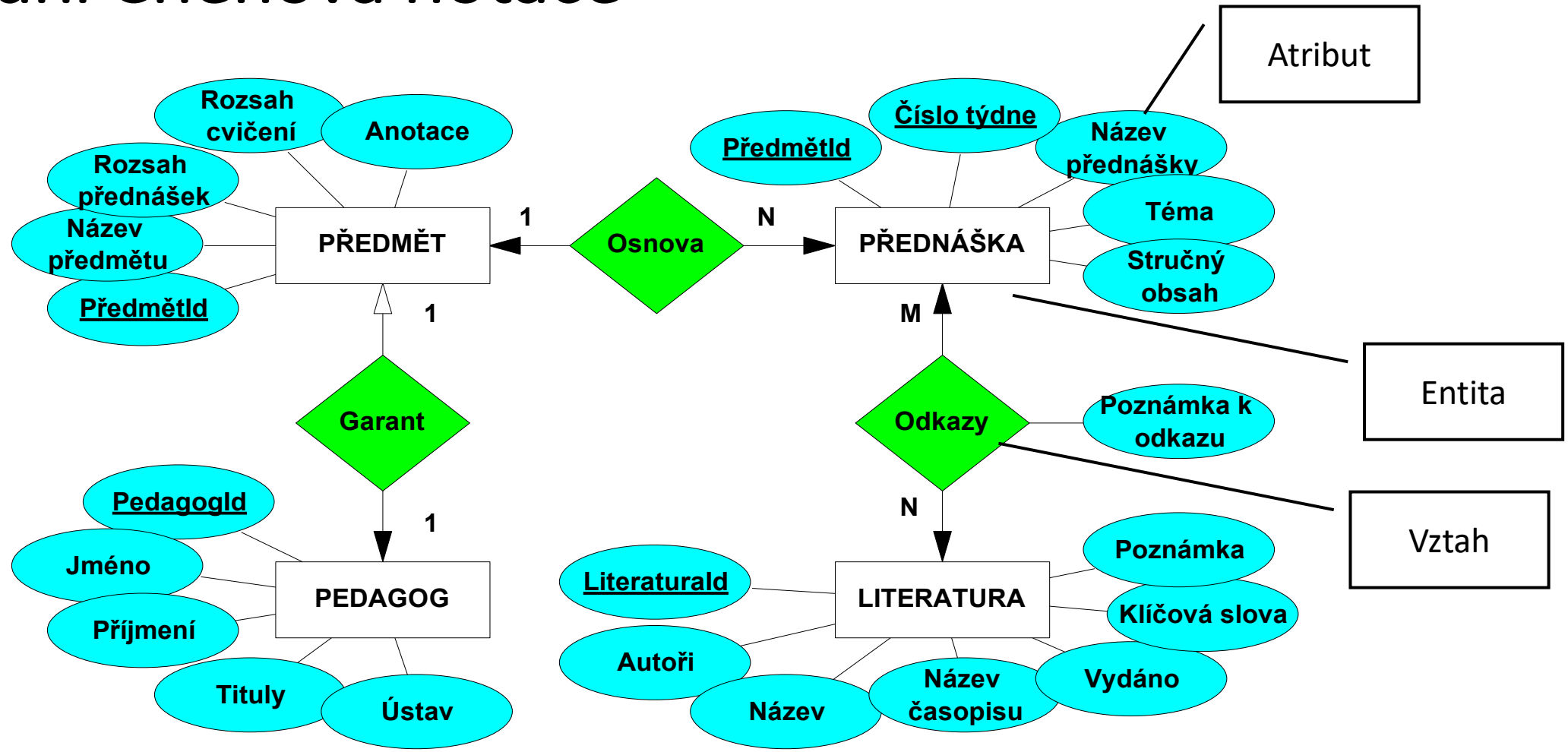
- ID závislé – klíč slabé entity je definován i s pomocí klíče silné entity, na které závisí. Bez výskytu silné entity nemůže existovat ani výskyt slabé entity a nemůže být ani samostatně identifikován. Je to silnější druh závislosti, než
- ID nezávislé (existenčně závislé) – mají vlastní klíč, takže k identifikaci silnou entitu nepotřebují, ale jejich existence (lépe řečeno existence výskytu entity) nemá bez existence výskytu silné entity smysl.

- Častým příkladem ID závislé slabé entity je vazební entita, pomocí které se realizuje dekompozice vztahů M:N na 1:N

Způsoby reprezentace

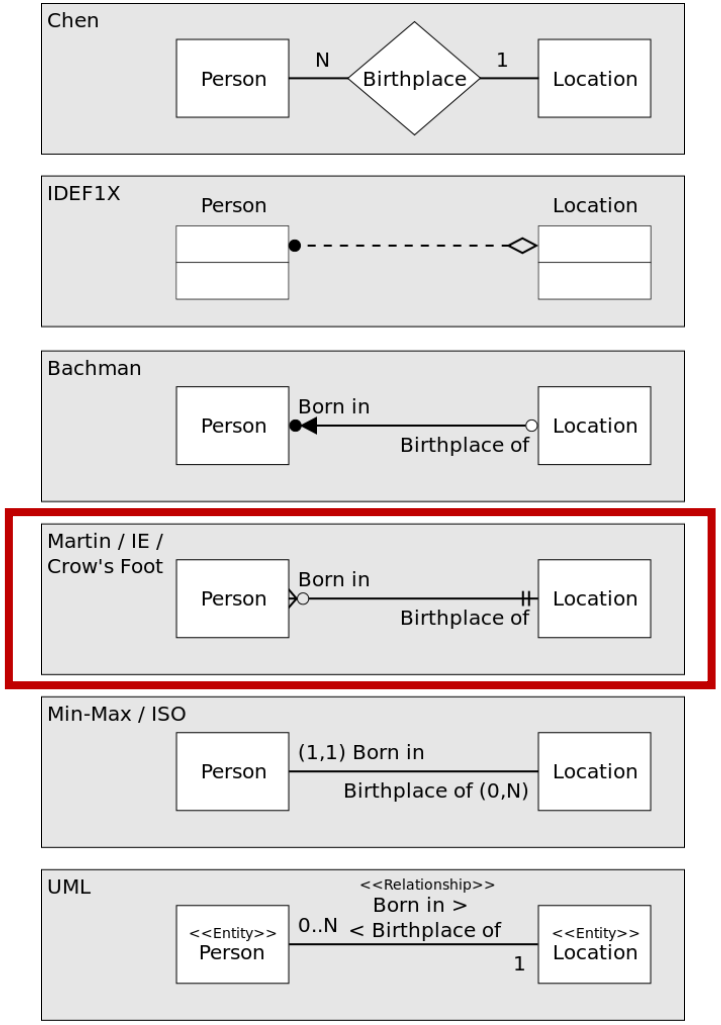
- Výsledkem ER modelování je obvykle grafické reprezentace modelu ve formě ER diagramu. (diagramu entit a vztahů)
- Existují různé možnosti zobrazení (notace), viz následující slidy

Původní Chenova notace



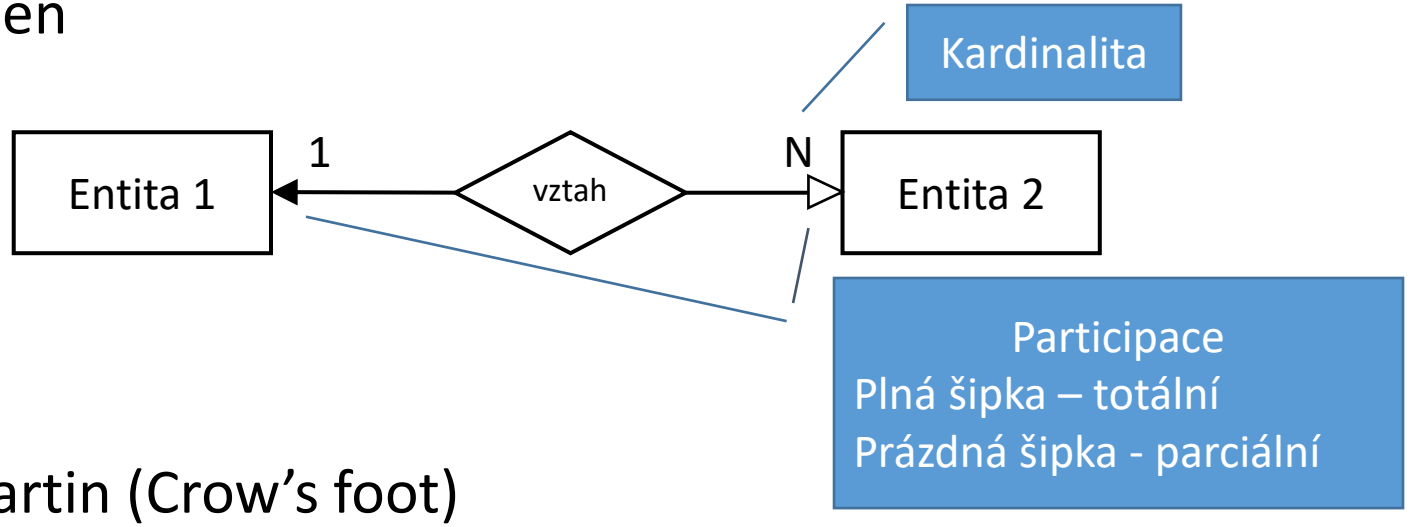
EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Alternativní notace



Vyjádření multiplicity vztahů

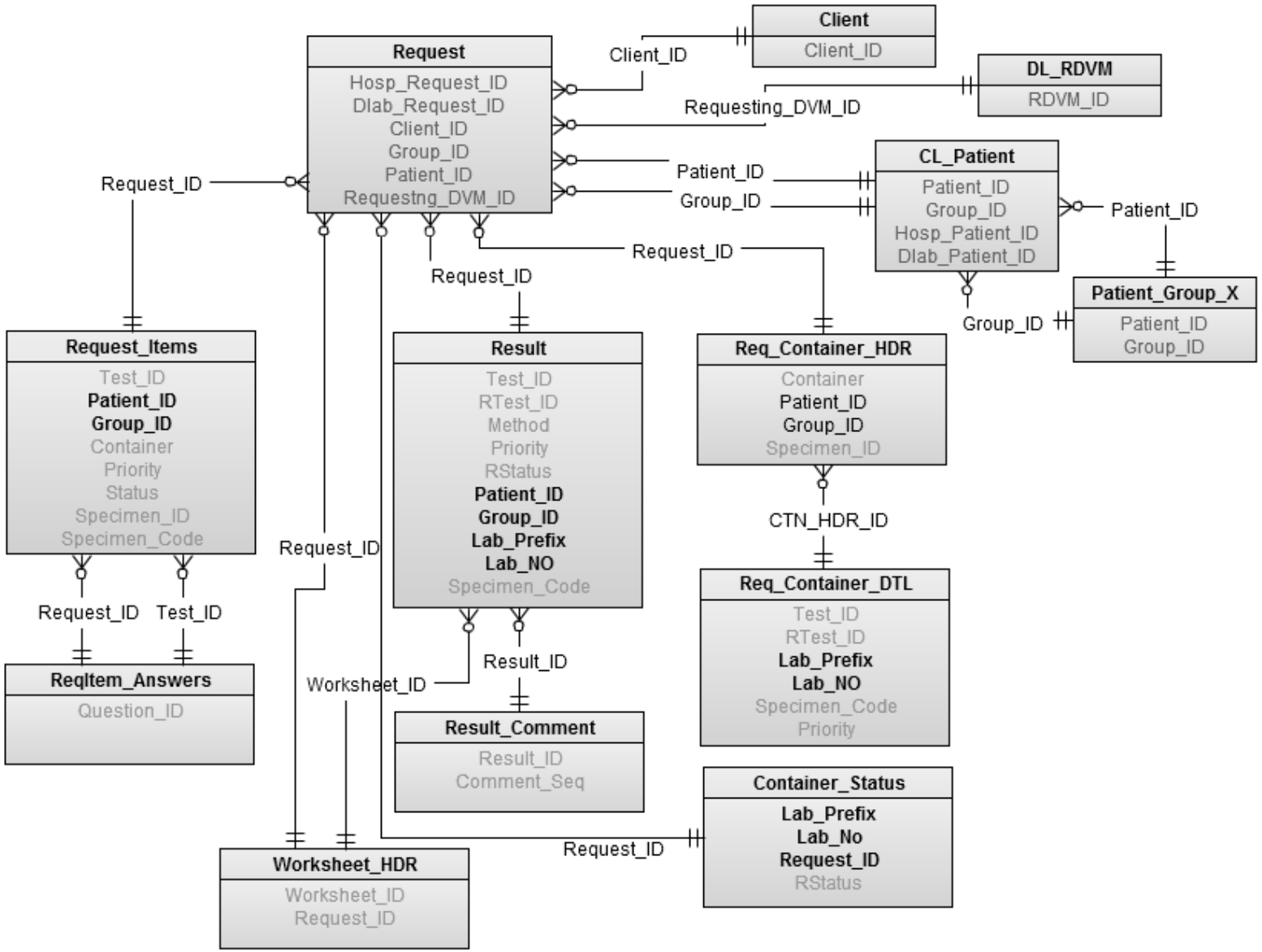
- Chen



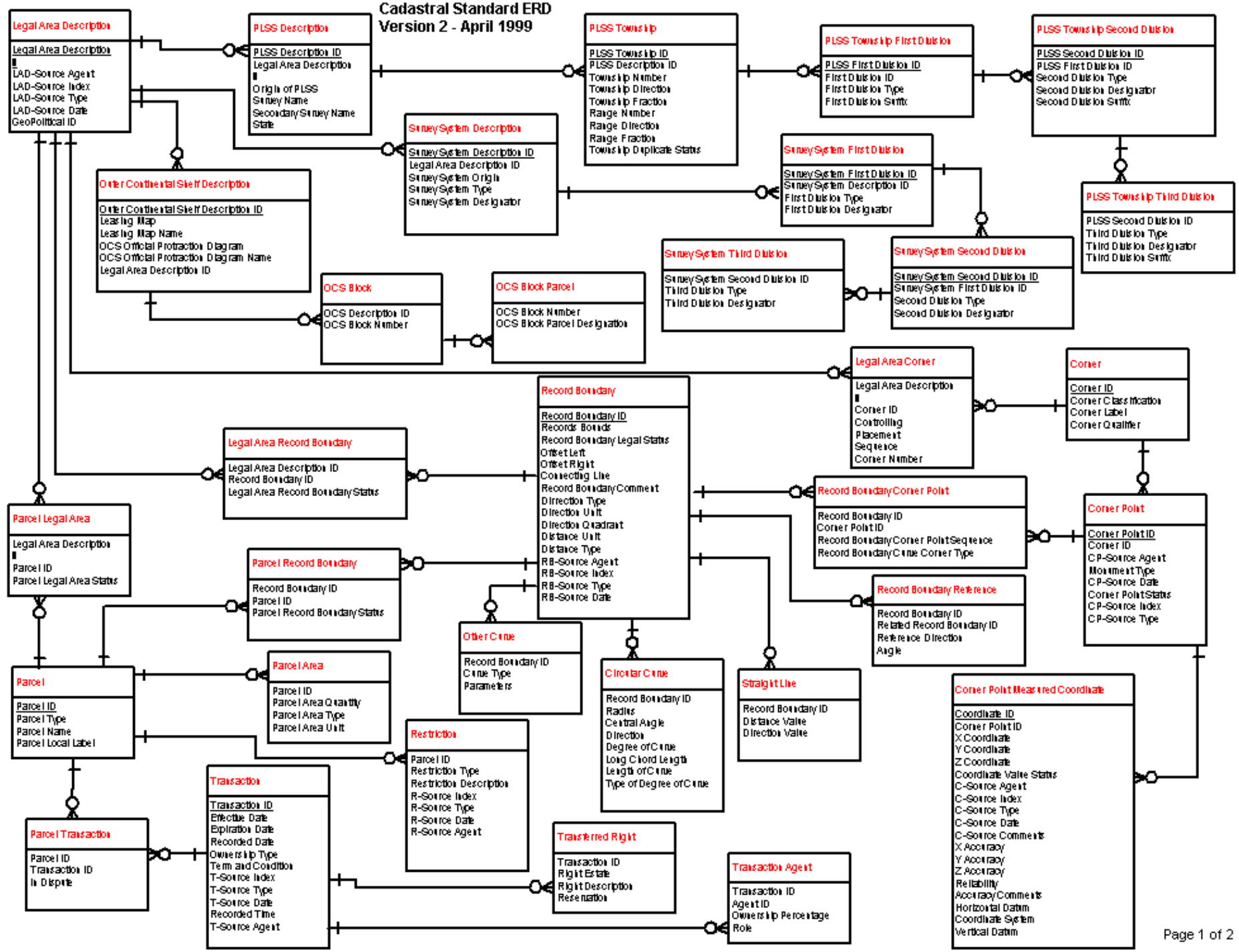
- Martin (Crow's foot)

Parciální 0..N | Totální 1..N | Totální 1..1 | Parciální 0..1

Clinpath Database Entity Relationship Diagram (ERD)



Cadastral Standard ERD
Version 2 - April 1999



Zdroje

- Kroenke, David a Auer, David J. Databáze. 1. vyd. Brno: Computer Press, 2015. 496 s. ISBN 978-80-251-4352-0.
- Sheldon, Robert. SQL: začínáme programovat. 1. vyd. Praha: Grada, 2005. 499 s. Průvodce. ISBN 80-247-0999-6.