VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

# Nástroj pro modelování relační databáze

# Relational Database Modeling Tool

2017 Radek Svoboda

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

# Zadání bakalářské práce

Student: **Radek Svoboda**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Nástroj pro modelování relační databáze
Relational Database Modeling Tool

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je vytvořit aplikaci s grafickým uživatelským rozhraním pro návrh schématu relační databáze formou E-R diagramu. Aplikace bude podporovat jak aktualizaci schématu databáze na základě změn v E-R diagramu, tak aktualizaci E-R diagramu podle existujícího schématu. Aplikace bude pracovat se SŘBD Microsoft SQL Server a Oracle Database.
Práce bude splňovat následující body:
1. Popis notace E-R diagramu.
2. Popis a srovnání stávajících CASE nástrojů pro modelování relačních databází.
3. Návrh a implementace grafického editoru E-R diagramů.
4. Návrh a implementace synchronizace E-R diagramů s relačním schématem.

Seznam doporučené odborné literatury:

[1] CHEN, Peter Pin-Shan. The entity-relationship model—toward a unified view of data. ACM Transactions on Database Systems (TODS), 1976, 1.1: 9-36.
[2] ULLMAN, Jeffrey D.; WIDOM, J. Database Systems: The Complete Book. 2000.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Petr Lukáš**

Datum zadání: 01.09.2016
Datum odevzdání: 28.04.2017


doc. Dr. Ing. Eduard Sojka
*vedoucí katedry*

prof. RNDr. Václav Snášel, CSc.
*děkan fakulty*

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 28. dubna 2017 .....

Rád bych na tomto místě poděkoval vedoucímu mé bakalářské práce Ing. Petru Lukášovi za poskytnuté rady a čas strávený konzultacemi.

**Abstrakt**

Tato bakalářská práce se zabývá návrhem a implementací softwarového nástroje pro návrh schématu relační databáze formou ER diagramu. Nástroj umožní uživateli nejen tvorbu nových schémat databáze, ale také vizualizaci již existujících schémat a jejich následné modifikace prostřednictvím změn ER diagramu v grafickém editoru. V první části se práce zabývá notací ER diagramu a srovnáním již existujících nástrojů určených k tvorbě ER diagramů. Druhá část se zaměřuje na návrh a implementaci jak grafického editoru, tak na problematiku synchronizace ER diagramu s existujícím relačním schématem.

**Klíčová slova**: relační databáze, ER diagram, relační schéma, CASE nástroj

**Abstract**

This thesis describes design and implementation of software tool for relational database schema design in form of an ER diagram. Tool is not limited only for creation of new database schemas, but it also provides visualization of already existing schemas and their subsequent modifications through changing ER diagram in graphic editor. The first part deals with ER diagram notation and the comparison of existing tools for the creation of ER diagrams. The second part focuses on the design and implementation of both the graphical editor, and on the synchronization of ER diagram with existing relational schema.

**Key Words**: relational database, ER diagram, relational schema, CASE tool

# Obsah

# Seznam použitých zkratek a symbolů

| | | |
|---|---|---|
| SQL | – | Structured Query Language |
| DDL | – | Data Definition Language |
| CASE | – | Computer-Aided Software Engineering |
| ERD | – | Entity-Relationship Diagram |
| DBMS | – | Database Management System |
| DSD | – | Data Structure Diagram |
| EER | – | Enhanced Entity–Relationship Model |
| IDEF | – | Integration Definition for Information Modeling |
| ICAM | – | Integrated Computer-Aided Manufacturing |
| JSON | – | JavaScript Object Notation |
| XML | – | Extensible Markup Language |

# Seznam obrázků

# Seznam tabulek

# Seznam výpisů zdrojového kódu

# 1 Úvod

Strukturované uložení dat je dnes jednou ze základních potřeb většiny vyvíjených aplikací. S rostoucím množstvím uchovávaných dat a potřebou jejich analýzy už dávno není dostačující prosté uložení do souboru. Na tyto potřeby reagují, dnes ve velké míře používané, relační databáze. Před samotným uložením dat je nutné definovat jejich strukturu, konkrétně relační model. Dnešní svět se velmi rychle vyvíjí, tedy mění, a tyto změny často vyvolají potřebu upravit datový model tak, aby co nejlépe reflektoval modelovanou realitu. Právě pro tuto činnost jsou dnes stále častěji používané CASE nástroje, které umožňují, nejen, snadnou změnu datového modelu prostřednictvím grafického uživatelského rozhraní bez nutnosti psaní vlastních DDL skriptů.

Vyvíjená aplikace slouží pro návrh a aktualizace schématu relační databáze, konkrétně podporuje DBMS Microsoft SQL Server a Oracle Database, z toho důvodu je první kapitola věnována popisu notace ER diagramu, který se používá pro vizualizaci vztahů mezi entitami v relačním schématu nejčastěji. Další kapitola se zaměřuje na srovnání možností již existujících nástrojů pro tvorbu ER diagramu. V následující kapitole je popsán návrh a implementace grafického editoru ER diagramu, včetně algoritmů pro vyhledávání cest při vizualizaci vztahů a porovnání jejich výkonosti. Poslední kapitola se zabývá synchronizací mezi ER diagramem a relačním schématem, zejména spoluprací použitých návrhových vzorů pro dosažení požadované funkcionality pro oba, dříve zmíněné, systémy řízení báze dat.

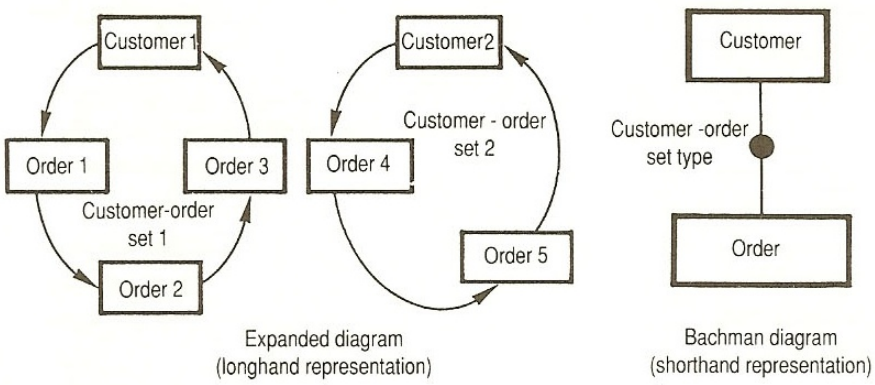
# 2 Popis notace ER diagramu

V kontextu relačních databázi jsou ER diagramy nejběžnějším způsobem vizualizace entit vyskytujících se v databázi a jejich vzájemných vztahů. V průběhu let vznikla celá řada notací, které se liší nejen použitými grafickými symboly pro znázornění jak entit, tak kardinality a povinnosti vztahů, ale i dalšími vlastnostmi.

Dodnes nepředstavuje žádná z notací standard v oblasti modelování databází, což je nejvíce patrné na celé řadě CASE nástrojů, které používají mnohdy svou vlastní notaci či dokonce kombinaci symbolů z několika různých notací.

## 2.1 Historie

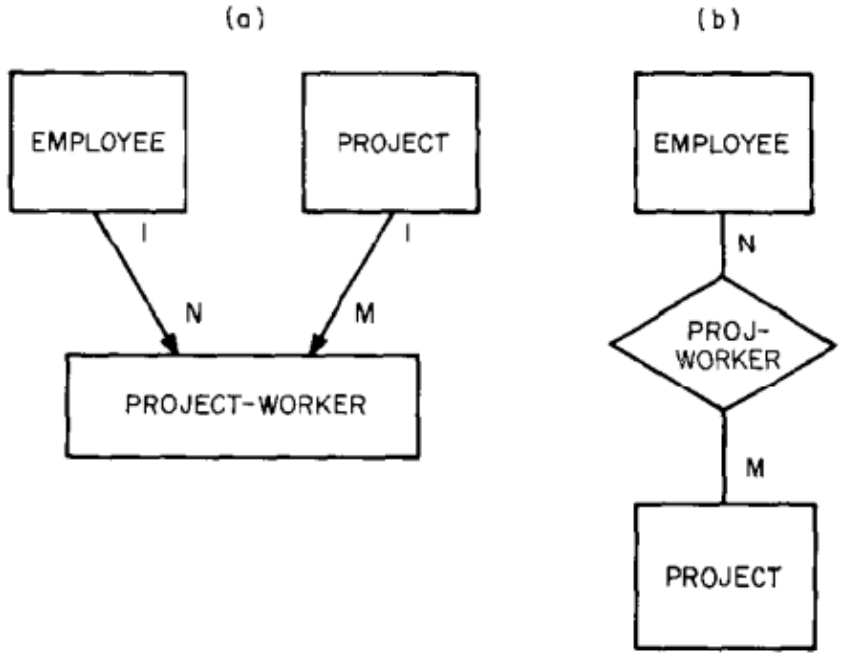
Za předchůdce ER diagramu jsou považovány diagramy datových struktur (DSD) [4], které rovněž slouží k zachycení entit, vztahů mezi nimi včetně vztažených omezení. Cílem DSD je graficky znázornit dekompozici složitých entit na jednotlivé elementy, k zachycení struktury dat se předpokládalo použití tzv. datových slovníků, které se dají zjednodušeně popsat jako množiny tabulek obsahující metadata, z tohoto principu vychází i systémové katalogy moderních DBMS.




Obrázek 1: Bachmanův diagram [5]

Speciálním typem DSD je Bachmanův diagram, který slouží k vytvoření relačního datového modelu bez ohledu na způsob fyzického uložení dat v systému. Tento diagram měl patrně největší vliv na pozdější vznik ER diagramu, který publikoval Peter Chen roku 1976 [1].

Největším rozdílem mezi DSD a ERD, opomineme-li grafickou podobu, je fakt, že DSD se zaměřuje na vztahy mezi jednotlivými elementy entit, zatímco ERD řeší vztahy mezi samotnými entitami.




(a) Data structure diagram (b) ER diagram

Obrázek 2: Srovnání DSD a ERD [1]

## 2.2 Využití ER diagramu

Nejčastější použití ER diagramu představuje tvorba nových relačních schémat při návrhu databází, ale rozhodně se nejedná o jedinou oblast využití. Tvorba ER diagramu velmi často stojí na počátku vývoje mnoha informačních systému, protože relační databáze jsou obvykle primárním datovým zdrojem těchto systému pro uložení dat transakčního charakteru.

ER diagramy umožňují zachytit vztahy a omezení entit v problémové doméně, proto jsou používány nejen při návrhu datové vrstvy, ale i v dalších fázích softwarového procesu. Doménový model aplikace je často ovlivněn právě ER diagramem relační databáze, protože na základě jejich odlišností jsme schopni odhadnout trivialitu objektově-relačního mapování. Trivialita tohoto mapování je často zásadní pro volbu vhodného návrhového vzoru pro implementaci vrstvy přístupu k datovému zdroji. Časté je i využití ER diagramu ve spojitosti s diagramy datových toků, pro znázornění uložení dat business procesu.

Dalším velmi častým použitím ER diagramu je vizualizace již existujících schémat databáze, protože právě grafické znázornění schématu je ideální pro snadné odhalení logických chyb, které mají za následek omezené možnosti aplikace, v některých případech i propady výkonu při náročnějších operacích s daty. Vizualizace existujících schémat nemusí být použitá pouze pro odhalení chyb, ale také při situaci, kdy došlo ke změně některých business procesu. Tyto změny často vyvolají nutnost přizpůsobit relační model změnám, ER diagram je tedy vhodný prostředek nejen

k analýze stávajícího schématu pro návrh změn, ale také pro provádění těchto změn použitím CASE nástrojů.

## 2.3 Prvky ER diagramu

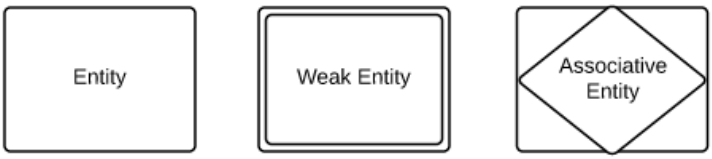
V této části jsou popsány prvky používané při tvorbě ER diagramu, včetně speciálních případů těchto prvků a příkladů. Ilustrační grafické znázornění odpovídá notaci podle P. Chena, nicméně v další části budou ostatní používané notace dále rozvedeny [7].

### 2.3.1 Entitní typ

Entitní typy (často označovány v souvislosti s ERD jen jako entity) jsou jednoznačně identifikovatelné objekty vyskytující se v problémové doméně, které jsou pro nás zajímavé svými atributy, daty, které chceme uchovávat např. *zákazník, oddělení, automobil.* Obvykle se označují podstatným jménem v jednotném čísle.

Rozlišujeme následující typy entit:

- **Silný entitní typ** - muže existovat nezávisle na ostatních entitních typech, protože obsahuje alespoň jeden atribut, který entitu jednoznačně odlišuje od ostatních, značí se obdélníkem
- **Slabý entitní typ** - jeho existence je závislá na jiném entitním typu, protože neobsahuje atribut umožňující jednoznačnou identifikaci, značí se obdélníkem s dvojitým okrajem
- **Asociativní entitní typ** - slouží k vyznačení vztahu mezi entitami, obsahuje atributy identifikující tento vztah, značí se kosočtvercem v obdélníku




Obrázek 3: Znázornění entit podle Chenovy notace [7]

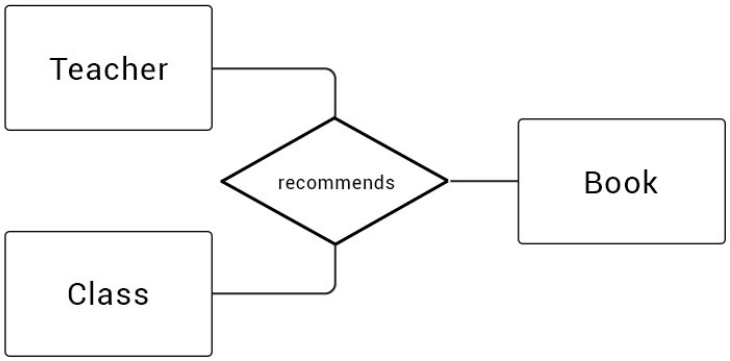
### 2.3.2 Vztah

Vztah slouží k vyjádření asociace mezi entitními typy, obecně vyjadřuje informace, které nelze vyjádřit pouhými entitními typy, např. student *studuje* předmět, novinář *napsal* článek. Označuje se slovesem. Znázorňuje se kosočtvercem.

Identifikující vztah je speciální typ vztahu, který spojuje slabou entitu s entitou, na které je závislá, ta se označuje jako vlastník. Značí se kosočtvercem s dvojitým okrajem.

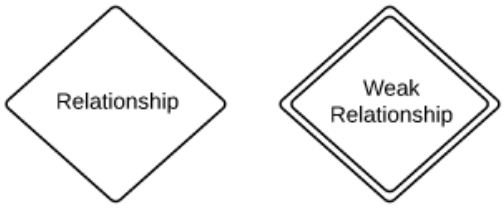
Vztahy dělíme podle počtu entit, které se v nich vyskytují na vztahy:

- **Unární** - ve vztahu vystupuje pouze jedna entita, ta je ve vztahu sama se sebou, např. zaměstnanec *je nadřízeným* jiného zaměstnance
- **Binární** - vztah mezi dvěma entitami, např. zaměstnanec *je vedoucím* oddělení
- **Ternární** - vztah mezi třemi entitami najednou, např. učitel *doporučuje* knihu třídě
- **N-ární** - vztah mezi $n$ entitami zároveň




Obrázek 4: Příklad ternárního vztahu

Obecně se ternární vztahy vyskytují v ER diagramech velmi málo, protože vedou k problémům s dekompozicí na několik binárních vztahů při přechodu z konceptuálního modelu na relační (logický), kdy může dojít ke ztrátě některých informací.




Obrázek 5: Znázornění vztahů podle Chenovy notace [7]

Vztahy dále dělíme podle kardinality, která udává, kolikrát může instance entitního typu být ve vztahu s instancí jiného entitního typu, pro binární typ vztahu rozlišujeme následující typy kardinality:

- **1:1**
- **1:N**
- **M:N**

Pro ternární typ vztahu rozlišujeme kardinalitu 1:1:1, 1:1:N, 1:N:M, a M:N:P, pro n-ární typy vztahu se typy kardinality odvozují analogicky.

Poslední důležitou vlastnosti vztahu je povinnost, určující zda může instance entitního typu existovat bez vztahu k jiné instanci entitního typu. Rozlišujeme povinné a nepovinné vztahy.

### 2.3.3 Atribut

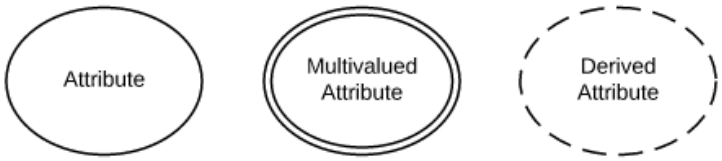
Atributy jsou vlastnosti charakterizující entitní typ. Označují se podstatným jménem, znázorňují se elipsou.

Rozlišujeme následující kategorie atributů:

- **Jednoduchý** - atomický atribut, nelze dále dělit, např. *značka automobilu*
- **Složený** - dělí se na více atomických atributů, např. *adresa* se dělí na *město, ulici a PSČ*
- **Odvozený** - hodnota je vypočtena na základě hodnot ostatních atributů, fyzicky nemusí v relační databázi existovat, znázorněn je elipsou s čárkovanou hranou, např. *průměrná mzda*

Další rozdělení je podle počtu hodnot atributu na:

- **Jednohodnotové** - atribut obsahuje pouze jednu hodnotu, např. *rodné číslo*
- **Vícehodnotové** - atribut může obsahovat více hodnot, značí se elipsou s dvojitým okrajem,např. *telefonní číslo na pevnou linku i mobilní telefon*




Obrázek 6: Znázornění atributů podle Chenovy notace [7]

Označení atributů lze kombinovat, např. *jednohodnotový odvozený atribut*

## 2.4 Kategorie modelů

Podle úrovně abstrakce rozlišujeme tři typy datových modelů [6], které využíváme při návrhu databázového schématu. Nejvyšší úroveň abstrakce nabízí konceptuální model, který zachycuje pouze entity a vztahy mezi nimi, je nejméně detailní. Více detailů nabízí logický model, poskytuje nižší úroveň abstrakce, ale stále je nezávislý na použitém DMBS. Posledním typem je fyzický model, ten vychází z logického modelu, nicméně se může v mnoha ohledech lišit, protože jeho cílem je obsáhnout dostatečný počet technických detailů pro implementaci databáze. Rozdíly

mezi logickým a fyzickým modelem způsobuje fakt, že fyzický model je vytvářen s ohledem na konkrétní použitou technologii, proto i v případě, že zvolíme relační databázi, se mohou pro různé DMBS fyzické modely více či méně lišit. Přehledné shrnutí rozdílů mezi modely nabízí tabulka 1.

Tabulka 1: Rozdíly mezi modely

| Modelované vlastnosti | Konceptuální | Logický | Fyzický |
|---|---|---|---|
| Názvy entit | ✓ | ✓ | ✗ |
| Vztahy | ✓ | ✓ | ✗ |
| Atributy | ✓ | ✓ | ✗ |
| Primární klíče | ✗ | ✓ | ✓ |
| Cizí klíče | ✗ | ✓ | ✓ |
| Názvy tabulek | ✗ | ✗ | ✓ |
| Názvy sloupců | ✗ | ✗ | ✓ |
| Datové typy sloupců | ✗ | ✗ | ✓ |

## 2.5 Vývoj notací

Stejně jako se vyvíjely potřeby návrhu datových modelů, vznikaly i nové notace ER diagramu. Cílem nově vznikající notací, zejména v průběhu 80tých let, bylo odstranění některých problémů již existujících notací.

Zásadním mezníkem pro vývoj notací byl objektově orientovaný přístup k vývoji softwaru, na který se mnohé notace snažily reagovat. Tento trend způsobil rozšíření notací ER modelu o koncept generalizace/specializace a dědičnosti, tento typ modelu se označuje jako Enhanced entity–relationship model (EER). Vlivem popularity objektově-relačních databází vznikly i notace umožňující v modelu zahrnout metody a operace entit. Tento přístup je dnes k vidění v celé řadě CASE nástrojů a vhodný je zejména pro modelování komplexních databází pro geografické informační systémy nebo telekomunikace.

## 2.6 Rozdíly notací

Na první pohled mohou různé notace vyvolávat dojem, že nejvýraznějším rozdílem mezi nimi je grafické znázornění entit a vztahů, nicméně rozdílů je mnohem více a některé mohou mít vliv na výslednou podobu fyzického modelu databáze.

### 2.6.1 Binární a n-ární modely

Nejzásadnějším rozdílem mezi notacemi je podpora n-árních modelů, z toho důvodů rozlišujeme dva typy modelů a to, binární a n-ární [3]. Pro kategorii binárních modelů je typické, že každý objekt, který má alespoň jeden atribut, je považován za entitu. Není tedy možné přiřadit atribut vztahu, jak je možné běžně vidět u Chenovy notace. Tento fakt je nejvíce patrný u vztahu

M:N, kterému chceme přiřadit atribut, pro zaznamenání dalších informací. Přidání neklíčového atributu způsobí nutnost přidat do modelu asociační entitu pro tento vztah.

V případě n-árních modelů není nutné vytvářet další entity k přidání atributů pro vztah. N-ární vztahy jsou z hlediska sémantiky výhodné pro modelování některých situací oproti vyjádření pomocí několika binárních vztahů. Při přechodu na modely nabízející nižší úroveň abstrakce, je často nutné tyto n-ární vztahy převést na sekvenci binárních vztahů, což může vést ke ztrátě cenný informací, protože už i pro ternární vztah M:N:P může způsobu dekompozice na binární vztahy existovat více a na první pohled nemusí být patrná ztráta části informace, kterou původní vztah nabízel. Tento problém může vést k logickým chybám při návrhu, které mohou mít později za následek velké množství změn v relačním schématu.

Podpora n-árních vztahů vede k nekompatibilitě mezi jednotlivými notacemi, kdy nemusí být snadné převést model používající jednu notaci na notaci jinou. Z tohoto důvodu dnešní CASE nástroje obvykle podporuji pouze jeden typ notace, případně typů podporují více, ale pouze ze stejné kategorie s ohledem na podporu arity vztahů.

### 2.6.2 Kardinalita a povinnost

Vyjádření kardinality a povinnosti vztahu je další z mnoha odlišností notací ER diagramu. Pro vyjádření kardinality se používá buď grafické znázornění, nebo označení číslem na obou stranách vztahů v případě starších notací, např. Chenova [2].

Povinnost bývá vyjádřena nejčastěji graficky, buď stylem čáry nebo určitým symbolem na stranách vztahu.

Speciálním vyjádřením je dvojice (min, max), která vyjadřuje jak kardinalitu, tak povinnost. Tento způsob je použit např. u Min-Max notace, která podle tohoto způsobu převzala i název.

Jednotlivé notace se neliší vyjádřením kardinality a povinnosti pouze graficky, případně textově, ale také tím, na které straně je informace vyjádřena. Rozlišujeme proto dva styly zápisu, první z nich se označuje jako look here, druhý jako look across [3].

Pokud modelujeme situaci, ve které jeden zaměstnanec pracuje v právě jednom oddělení a oddělení může mít několik zaměstnanců, v look across notaci zapíšeme 1 na stranu oddělení a N na stranu zaměstnance. Pro notaci look here tomu bude přesně naopak, tedy 1 zapíšeme na stranu zaměstnance a N na stranu oddělení. To, na které straně je vyjádřena kardinalita a povinnost nemusí být pro notaci jednotné, Chenova notace používá look across přístup pro vyjádření kardinality, ale povinnost vyjadřuje stylem look here, oproti tomu u notace Min-Max se obvykle dodržuje pouze jeden způsob pro kardinalitu i povinnost.

## 2.7 Nejčastěji používané notace

Jak už bylo zmíněno, notací existuje celá řada. Jednotlivé notace se v mnoha směrech liší, v následující kapitole jsou vybrané notace popsány, včetně příkladu. Příklad modeluje situaci, kdy

jeden člověk má právě jedno místo narození a na jednom místě narození se mohlo narodit více lidí.

### 2.7.1 Chenova notace

Jedná se o notaci představenou poprvé roku 1976 Peterem Chenem a dodnes se jedná o jednu z nejvíce používaných. Tato notace podporuje n-ární vztahy, původní specifikace obsahovala prvky jen pro entity, vztahy, včetně kardinality, a atributy. Později byla rozšířena o rozlišení povinnosti vztahu a koncept generalizace/specializace. Pro povinnost se používá styl look here, pro kardinalitu look across. Grafické znázornění jednotlivých prvků této notace je popsáno v kapitole 2.3.


Obrázek 7: Příklad Chenovy notace [6]

### 2.7.2 Bachmanova notace

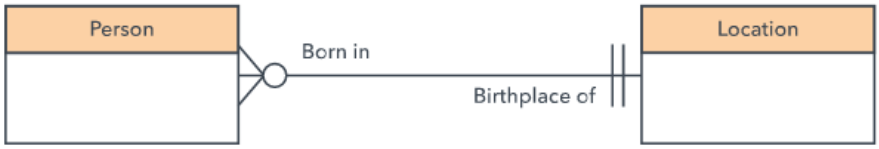
Tento typ notace podporuje nejvýše binární vztahy, vyjádření kardinality používá styl look across, povinnost pak look here, stejně jako je tomu u Chenovy notace. Bachmanova notace modeluje cizí klíče už na konceptuální úrovní, její předpokládané použití je tedy modelování relačních databází. Entity jsou znázorněny obdélníky, vztahy čárami. Vztahům není možné přiřadit atributy, pokud potřebujeme u vztahu M:N přidat neklíčový atribut, je nutná dekompozice pomocí asociační entity. Povinnost vztahu vyjadřuje kruh bez výplně, pro nepovinné vztahy, povinný vztah je naopak vyjádřen plným černým kruhem. Povinnost je vyjádřena vždy na konci vztahu. Pokud čára končí šipkou, jedná se o kardinalitu N, 1 se značí pouhou čárou bez symbolu.


Obrázek 8: Příklad Bachmanovy notace [6]

### 2.7.3 Crow's foot notace

Tato notace se označuje také jako Information engineering nebo Martinova notace. Opět podporuje nejvýše binární vztahy, které jsou znázorněny čárami. Není možné vztahu přiřazovat

atributy bez asociační entity. Kardinalita i povinnost vztahu používá look across styl a obojí je znázorněno graficky. Entity jsou znázorněny obdélníky, do nich jsou, kromě samotného názvu, vepsány také všechny atributy. Zajímavé je, že ačkoli každá notace používá své grafické symboly, tak znaky pro znázornění kardinality a povinnosti použité touto notací najdeme velmi často v kombinaci s jinými notacemi, jako je např. IDEF1X, detailní popis nalezneme v kapitole 2.7.6. Tato notace je populární zejména v oblasti CASE nástrojů a nalezneme ji v mnohých z nich.




Obrázek 9: Příklad Crow's foot notace [6]

### 2.7.4 Barkerova notace

Barkerova notace je velmi populární v oblasti tvorby datových modelů pro Oracle Database, vznikla roku 1981 a po příchodu Richarda Barkera do Oraclu, se stala v této oblasti velmi populární. Jedná se o notaci podporující nejvýše binární vztahy, které jsou vyjádřeny čárami. Plná čára symbolizuje povinný vztah, čárkovaná pak vztah nepovinný. Pro vyjádření kardinality je zde použít také symbol vraní nohy. Povinnost používá styl look here, kardinalita pak styl look across. Atributy jsou zde trojího typu, každý typ označuje kategorii atributu.

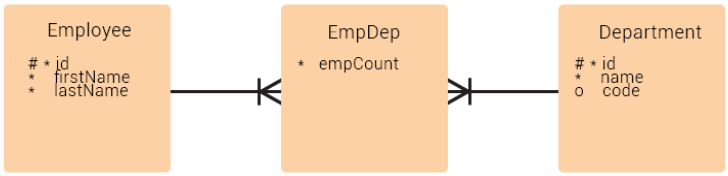
Notace používá tyto znaky pro kategorizaci atributů:

- # - identifikující atribut
- ∗ - povinný atribut
- ○ - nepovinný atribut


Obrázek 10: Příklad Barkerovy notace [6]

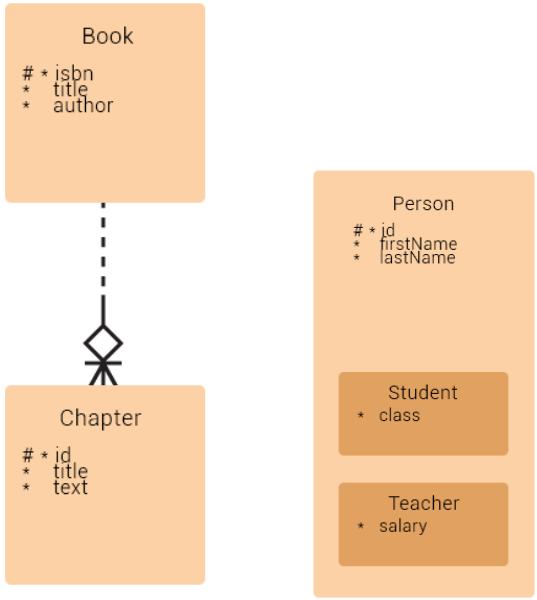
Oproti ostatním notacím zde nalezneme určitá specifika. Prvním z nich je tzv. UID čára, obr. 11, která se kreslí pouze u slabých entitních typů a vyjadřuje, že je identifikován atributy silných entitních typů.

Dalším specifikem, je možnost vyznačit nepřenositelný vztah, obr. 12a, pokud jej použijeme, už jej nelze později změnit, příkladem je vztah mezi kapitolou a knihou, kdy kapitolu z jedné




Obrázek 11: Specifika Barkerovy notace - UID

knihy nemůžeme přiřadit knize druhé. Graficky se tento typ vztahu značí přidáním kosočtverce k vraní noze.




(a) Nepřenositelný vztah (b) Podtypy

Obrázek 12: Specifika Barkerovy notace

Notace podporuje také dědičnost, kdy podtypy dědí atributy supertypu, obr. 12b, graficky je tento jev vyjádřen přidání obdélníků jednotlivých podtypů do společného obdélníku odpovídajícímu supertypu.

### 2.7.5 Min-Max notace

Jedná se o notaci, která podporuje nejvýše binární vztahy. Pro vyjádření kardinality a povinnosti vztahu, zde není použit žádný grafický symbol, ale uspořádaná dvojice $(min, max)$, kde minimální počet instancí entitního typu, které musí ve vztahu participovat, určuje povinnost, maximální počet určuje kardinalitu. Styl pro kardinalitu a povinnost se používá jednotný, zpravidla look here.


Obrázek 13: Příklad Min-Max notace [6]

### 2.7.6 IDEF1X notace

IDEF1X je notace, která vnikla v rámci programu ICAM, financovaném US Air Force, jako součást IDEF technik pro modelování informačních systémů. Vztahy jsou podporované nejvýše binární a neklíčové atributy vztahu je nutné modelovat použitím asociační entity. Kardinalita a povinnost vztahu je vyjádřena pomocí $(min, max)$ notace, buď graficky, nebo textově, a nejčastěji používá styl look across. Tato notace rozlišuje graficky silné entitní typy, značí se obdélníkem, a slabé entitní typy, značí se obdélníkem se zakulacenými rohy. Podobně jako u Barkerovy notace je zde možné vyznačit identifikující vztahy. Navzdory tomu, že notace nabízí vlastní grafické symboly pro znázornění kardinality a povinnosti vztahu, je často použita pro tento účel Crow's foot notace, zbytek prvků IDEF1X je ponechán.


Obrázek 14: Příklad IDEF1X notace [6]

## 2.8 Nevýhody ER diagramů

ER diagramy jsou určeny primárně pro vytváření modelů relačních struktur. Pokud pracujeme s nestrukturovanými daty, které není jednoduše možné reprezentovat relacemi, není ER diagram vhodný nástroj k vizualizaci jejich struktury. Toto omezení platí i pro částečně strukturovaná data, nejznámějšími zástupci jsou XML nebo JSON, protože navzdory tomu, že jsme schopni rozlišit jednotlivé entity, může dojít k situaci, kdy entity téhož typu se liší svými atributy.

## 2.9 Shrnutí

Každá notace má své klady i zápory, nelze tedy říci, že je některá notace je lepší než jiná, volba notace by měla vycházet z potřeb pro tvorbu konkrétního modelu. Největší rozdíl mezi notacemi představuje možnost modelování ternárních vztahů, případně i vztahů s vyšší aritou. Proto je nutné zvážit zda sémantické možnosti binárních notací jsou pro vytvářený model dostatečné.

Znázornění kardinality a povinnosti se u jednotlivých notacích liší spíše graficky a všechny zmíněné notace poskytují v této oblasti takřka identické možnosti. Mezi poslední parametry, které mohou volbu notace ovlivnit, řadíme možnost modelování cizích klíčů na konceptuální úrovni, dojde sice k snížení úrovně abstrakce, nicméně konverze na fyzický model je poté jednodušší. Posledním kritériem je možnost modelování subtypů a nepřenositelných vztahů. Toto kritérium, ale nabídku notací velmi omezí.

Přehledné srovnání vybraných notací nabízí tabulka 2.

Tabulka 2: Srovnání vybraných notací

| Vlastnost | Chen | Bachman | Crow's foot | Berker | Min-Max | IDEF1X |
|---|---|---|---|---|---|---|
| Ternární vztahy | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Look-across kardinalita | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ |
| Look-across povinnost | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ |
| (Min, Max) notace vztahů | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ |
| Neklíčové atributy vztahů | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| Cizí klíče na koncept. úrovni | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| Podtypy | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |

# 3 Popis a srovnání stávajících CASE nástrojů pro modelování relačních databází

V dnešní době je využití CASE nástrojů v mnoha fázích softwarovém procesu de facto standardem. Jinak tomu samozřejmě není ani u návrhu datových modelů. Na trhu nalezneme celou řadu nástrojů, od komerčního softwaru po open-source projekty, podporujících tvorbu ER diagramů.

Tyto nástroje z pravidla nenabízí pouze grafický editor pro vytváření těchto diagramů pro programovou dokumentaci, ale obsahují řadu dalších užitečných funkcí, jako je např. generování patřičných DDL skriptů na základě vytvořeného diagramu.

## 3.1 Výhody a nevýhody CASE nástrojů pro modelování relačních databází

Jednou z největších výhod je urychlení vývoje rozsáhlých projektů, protože mnoho činností lze použitím CASE nástrojů automatizovat. Mnoho z nich se nezastavuje u pouhého generování skriptů pro nasazení nové databáze, ale využívá reverzního inženýrství k vygenerování ER diagramu z existující databáze nebo na základě DDL skriptu. Stejně tak lze některé nástroje využít k tvorbě objektově-relačního mapování pro přístup k datovému zdroji, což ušetří mnoho času a práce. Tyto možnosti obvykle vedou ke značnému zvýšení produktivity.

Použití CASE nástrojů s sebou nenese jen samá pozitiva, ale skýtá i řadu nedostatků. Největším problémem je obvykle nulová podpora pro provedení formální analýzy vytvořeného modelu, nástroje tedy plně spoléhají na odborné znalosti uživatele.

Dalším problémem je časová investice nutná k naučení se efektivně pracovat s daným nástrojem, tento problém se netýká pouze modelování relačních databází, ale projevuje se u většiny specializovaných nástrojů.

Za poslední nevýhodu považuji cenu komerčních nástrojů, protože zejména u menších projektů nemusí být využity veškeré možnosti, které daný produkt nabízí a týmové licence mohou být velmi drahé. Řešením je použití open-source nástrojů, případně tzv. community verze některého z nástrojů, pokud to licenční politika dovolí.

## 3.2 Srovnání stávajících nástrojů

### 3.2.1 Oracle SQL Developer

### 3.2.2 Toad Data Modeler

### 3.2.3 SQL Server Management Studio

### 3.2.4 Visual Paradigm

### 3.2.5 StarUML

### 3.2.6 Online nástroje

## 3.3 Shrnutí

# 4 Návrh a implementace grafického editoru ER diagramů

# 5 Návrh a implementace synchronizace ER diagramů s relačním schématem

# 6 Závěr

# Literatura


[1] CHEN, Peter Pin-Shan. *The entity-relationship model—toward a unified view of data. ACM Transactions on Database Systems* [online]. 1(1), 9-36 [cit. 2017-03-10]. Dostupné z: http://portal.acm.org/citation.cfm?doid=320434.320440

[2] KRÁTKÝ, Michal a Radim BAČA. *Databazové systémy* [online]. [cit. 2017-03-11]. Dostupné z: http://dbedu.cs.vsb.cz/SubPages/OpenFile.aspx?file=book/dbcb.pdf

[3] SONG, Il-Yeol, Mary EVANS a E.K. PARK. *A Comparative Analysis of Entity-Relationship Diagrams* [online]. [cit. 2017-03-11]. Dostupné z: http://www.cci.drexel.edu/faculty/song/publications/p_Jcse-erd.PDF

[4] BACHMAN, C. W. *Data structure diagrams* [online]. [cit. 2017-03-11]. Dostupné z: http://www.minet.uni-jena.de/dbis/lehre/ws2005/dbs1/Bachman-DataStructureDiagrams.pdf

[5] Data structure diagram. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-03-11]. Dostupné z: https://en.wikipedia.org/wiki/Data_structure_diagram

[6] Lucidchart. *What is an Entity Relationship Diagram* [online]. [cit. 2017-03-10]. Dostupné z: https://www.lucidchart.com/pages/er-diagrams

[7] Lucidchart. *ER Diagram Symbols and Notation* [online]. [cit. 2017-03-10]. Dostupné z: https://www.lucidchart.com/pages/ER-diagram-symbols-and-meaning

# A Použité knihovny třetích stran

- MetroLib
- AvalonDock

# B Struktura přiloženého optického média

**Složka** Popis

# C Instalace programu

Popis instalace.