Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

E-R model dat

2015

Tomáš Kukučka

Vedoucí práce: Mgr. Jiří Zacpal, Ph.D.

Studijní obor: Aplikovaná informatika, prezenční forma

Bibliografické údaje

Autor:	Tomáš Kukučka
Název práce:	E-R model dat
Typ práce:	bakalářská práce
Pracoviště:	Univerzita Palackého v Olomouci, Přírodovědecká fakulta-Katedra informatiky
Rok obhajoby:	2015
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	Mgr. Jiří Zacpal, Ph.D.
Počet stran:	29
Přílohy:	1 CD/DVD
Jazyk práce:	český

Bibliograhic info

Author:	Tomáš Kukučka
Title:	Thesis title
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2015
Study field:	Applied Computer Science, full-time form
Supervisor:	Mgr. Jiří Zacpal, Ph.D.
Page count:	29
Supplements:	1 CD/DVD
Thesis language:	czech

Anotace

Dokument popisuje problematikou návrhu databáze pomocí ER modelu v různých notacích, dále převod do relačního modelu a do jazyka SQL. Problematika je demonstrována na aplikaci ErCreator, která umožňuje pohodlné vytváření ER diagramů.

Synopsis

Document describes database design using ER model in various notations, then transfer to the relational model and SQL language. It is demonstrated in application ErCreator. Is is tool to create ER diagrams.

Klíčová slova: E-R model; Chenova notace; Crow's foot notace; Relační model, SQL

Keywords: E-R model; Chen notation; Crow's foot notation; Relation model, SQL

Děkuji, panu doktorovi Mgr. Jiřímu Zacpalovi, Ph.D. za přípomínky a pomoc při tvorbě aplikace.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Seznam vět

Seznam zdrojových kódů

1 Úvod

V posledních letech se dostávají databáze do všech lidských odvětví. Databáze mohou být jednoduché například telefonní seznam, středně složité jako například internetový obchod, ale i extrémně složité, například databáze výrobní linky průmyslové společnosti. Je proto nesmírně důležité databáze správně navrhnout, aby struktura databáze byla efektivní a zároveň srozumitelná. V následujícím textu se podíváme na několik způsobů návrhu databáze a vše si budeme demonstrovat na aplikaci ErCreator.

1.1 Požadavky na aplikaci

Aplikace bude určena pro vytváření E-R modelů při výuce databází. Aplikace nebude vyžadovat žádnou znalost databází. Primárně bude určena pro výuku, ale bude ji možné použít pro návrh vlastní databáze. Aplikace by měla splňovat následující cíle.

- Aplikace by měla umožňovat pohodlné vytváření E-R modelů.
  - Aplikace bude umožňovat vytváření E-R modelů v několika notacích.
- Aplikace by měla umět převádět E-R diagramy do jazyka SQL.
- Součástí aplikace by měla být i výuková část, která by popisovala problematiku různých datových modelů.
- Poslední částí aplikace by měla být část testovací, která by vhodnou metodou testovala uvedenou problematiku.

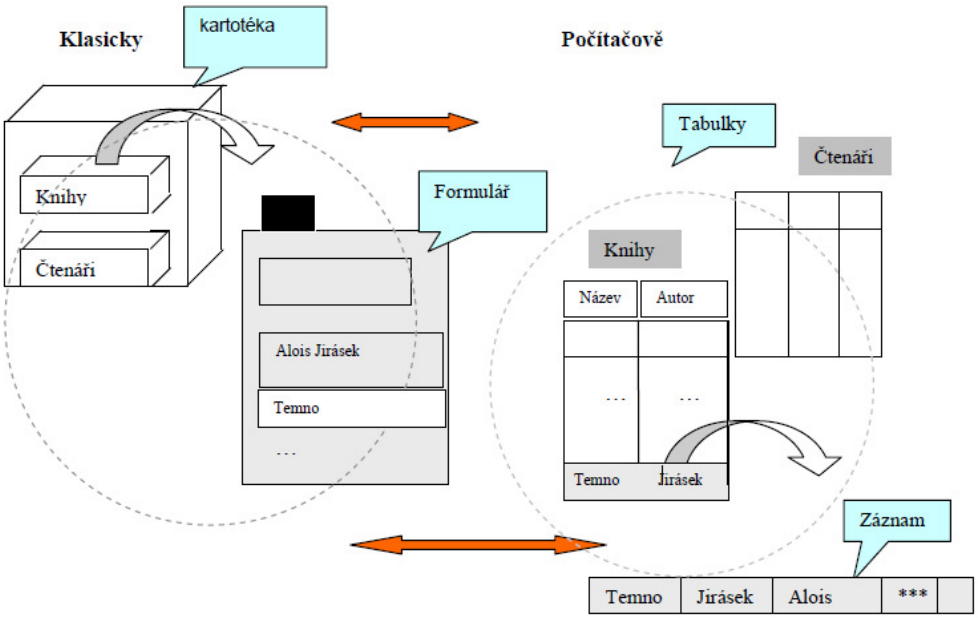
1.2 Stručný popis kapitol

V následující kapitole se podíváme na teorii a na problematiku databázových modelů a notací Ve třetí kapitole si ukážeme problematiku při vytváření aplikace ErCreator a rozebereme si její implementaci. Ve třetí čtvrté si představíme aplikaci ErCreator, popíšeme si všechny její funkce. V předposlední kapitole se podíváme na podobné aplikace. Poslední kapitola bude závěr.

2 Úvod do databázové technologie

2.1 Databáze

Sdílená kolekce logicky souvisejících dat i s popisem své datové struktury, organizovaná pro optimální manipulaci s perzistentními daty a získávání informací pro potřeby informačního systému. Pro základní představu porovnejme zjednodušené analogie klasické a elektronické verze na příkladu kartotéky části knihovny (je použit relační model dat, který data uchovává ve formě tabulek)[1].



Obrázek 1: Porovnání klasické a elektronické technologie [1]

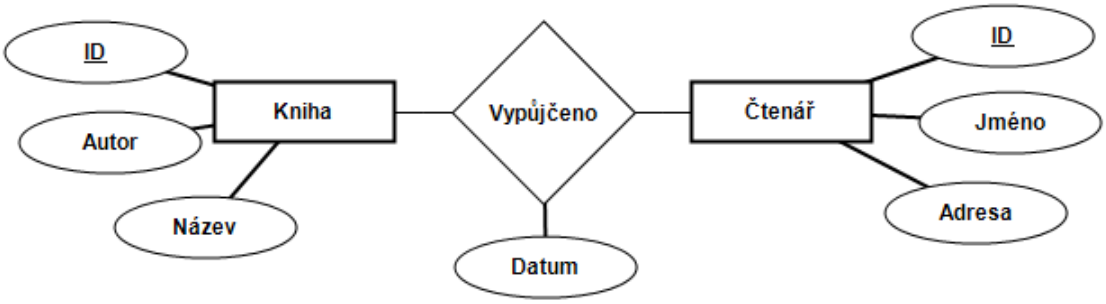
Neformálně lze databázi chápat jako množinu pojmenovaných datových tabulek.

Hlavní funkce databáze:

- Definice dat - definujeme, jaká data budou v databázi (čísla, text, datum, multimediální data).
- Manipulace s daty - s daty můžeme libovolně manipulovat, přidávat, odstraňovat, měnit.
- Řízení dat - můžeme nastavit, kdo bude moci data číst i měnit, pouze číst a kdo k nim nebude mít vůbec přístup.

Charakteristika dat v databázi[1]:

- Perzistence – data přetrvávají dlouhodobě od jedné operace ke druhé, nezávisle na použitých programech.
- Velké množství – operace typicky nevystačí s vnitřní pamětí, proto použití sofistikovaných algoritmů při manipulaci s daty.
- Správnost, nerozpornost – snaha odhalením nejrůznějších chyb v datech při vkládání nebo úpravě databáze zachovat korespondenci s realitou, vztaženou ke konkrétnímu času, ne nutně k nejaktuálnějšímu (realizováno pomocí integritních omezení).



Obrázek 2: Ukázka ER-Diagramu, jednoduchý příklad knihovny

- Spolehlivost – data je možné po poruše počítače zrekonstruovat.
- Sdílení – s daty pracuje typicky více uživatelů.
- Bezpečnost – možnost omezit přístup k datům a operacím s nimi.
- Integrace – spojení několika požadovaných pohledů do komplexní datové struktury.
- Konzistence – identická data mohou být dočasně nebo trvale uložena na více místech, ale musí mít stejnou hodnotu.

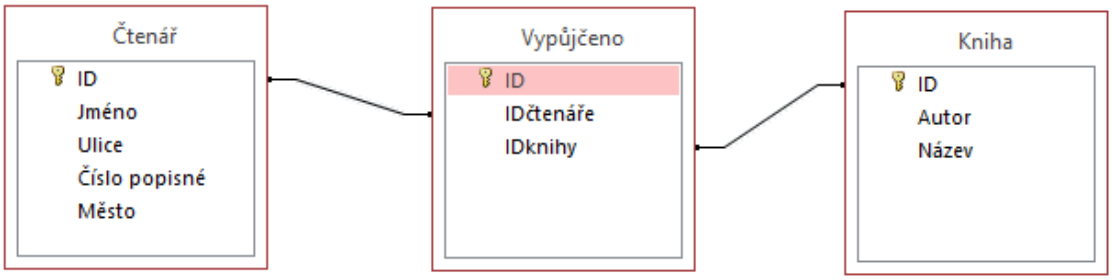
2.2 Model dat

Množina abstraktních a soběstačných formálních definic datových struktur a operací s daty (případně dalších operací, omezení a podobně), které dohromady tvoří formální výpočetní model, se kterým mohou uživatelé interagovat[2].

Jinými slovy, datový model popisuje sémantiku, strukturu, vztahy, integritní omezení na základě určité abstrakce.

Možné rozdělení datových modelů:

- Konceptuální - největší abstrakce, popisuje data nezávisle na jejich uložení. Je výsledkem datové analýzy. Nejznámější je ER model. Příklad obrázek 2.
- Logický - menší míra abstrakce, již pracujeme s daty závisle na tom, jak budou uložena, uvádíme datové typy. Nejznámější je Relační model. Příklad obrázek 3.
- Fyzický - nejmenší míra abstrakce, zde vybíráme konkrétní databázovou platformu, data jsou uložena fyzicky na disku.



Obrázek 3: Ukázka relačního modelu, jednoduchý příklad knihovny

2.3 Relační model

Relační model je datový model, který se stal konceptuálním základem relačních databází, tento model je založen na relačních množinách a predikátové logice. Model byl poprvé formulován E. F. Coddem.

Přínosem relačního modelu dat je zejména oddělení logické struktury dat od implementace, transparentnost přístupových metod při manipulaci s daty a matematická podpora pro omezení redundance pří návrhu struktury databáze.

Hlavní složky relačního modelu dat:

- Relační datová struktura - v relačním modelu jsou veškerá data a vztahy mezi nimi uložena tabulkách, které se skládají z řádků a sloupců. Sloupce musí mít definovaný název, rozsah a datový typ - souhrnně nazýváno doména. Řádky pak zastupují jednotlivé datové n-tice.

- Integritní omezení - obecná integritní jsou omezení, která vyplývají z reality dat reprezentovaných v databázi.

- Kandidátní klíč - kandidátní klíč je jeden z atributů, který je jednoznačný a neredukovatelný.

- Primární klíč - v každé relaci musí existovat primární klíč, který jednoznačně označuje n-tici v rámci relace (musí být unikátní v rámci tabulky). Primárním klíčem je jeden z kandidátních klíčů a může se sestávat jak z jediného atributu, tak může být složen z více atributů. Pokud v praxi není možné takovýto atribut vybrat, je přidán atribut “Id”, který každému řádku přidělí rámci tabulky neopakovatelné číslo.

- Alternativní klíč - kandidátní klíč, který není primárním klíčem.

- Cizí klíč - slouží k propojení různých tabulek, které mají něco společného. Cizí klíč je sloupec tabulky, který odpovídá primárnímu klíči jiné (nebo téže) tabulky. Propojením tabulek přes cizí klíč se mezi tabulkami vytváří vazby 1:1, 1:N, M:N.

  - 1:1 - jeden záznam v první tabulce má vazbu s právě jedním záznamem z druhé tabulky. Každý zaměstanec má právě jedne počítač. Každý počítač má právě jednoho zaměstance.
  - 1:N - jeden záznam v první tabulce může být propojen s více záznamy z druhé tabulky. Zaměstanec může mít více než jeden počítač. Každý počítač má pravě jednoho zaměstance.
  - N:M - v první tabulce může být více záznamů, které jsou propojeny s více záznamy z druhé tabulky. Zaměstanec může mít více než jeden počítač. Počítač může mít více zaměstanců.

- Vazební tabulka - u vazby M:N není možné docílit správného propojení záznamů. K propojení této vazby se využívá vazební tabulka, která se skládá z primárních klíčů propojovaných tabulek.

Pravidla:

- Pravidlo integrity entit - u žádné součásti primárního klíče nesmí chybět hodnota, z čehož vyplývá, že každá n-tice musí být jednoznačné identifikována hodnotou primárního klíče.

- Pravidlo referenční intergrity - databáze nesmí obsahovat nesouhlasnou hodnotu cizího klíče.

2.4 SQL (Structured query language)

Standardizovaný jazyk pro komunikaci s databázemi. Součástí databázového jazyka musí být prostředky Jazyk pro specifikaci schématu (DDL - Data Definition Language).

Slouží k vytvoření tabulkové struktury, tedy vytvoření tabulek, atributů, datových typů. Mezi nejdůležitější patří příkazy uvedeny níže.

- create - příkaz sloužící k vytvoření tabulky/sekvence.

- drop - příkaz sloužící ke zrušení tabulky/sekvence.

- alter - příkaz sloužící k úpravám tabulky

Jazyk pro manipulaci s daty - nebo-li DML - Data Manipulation Language. Obsahuje prostředky pro databázové operace (např. vyhledávání, mazání, vkládání,...). Mezi nejdůležitější patří příkazy uvedeny níže.

- select - slouží k výběru záznamů z databáze (možnost i na základě kritérií).

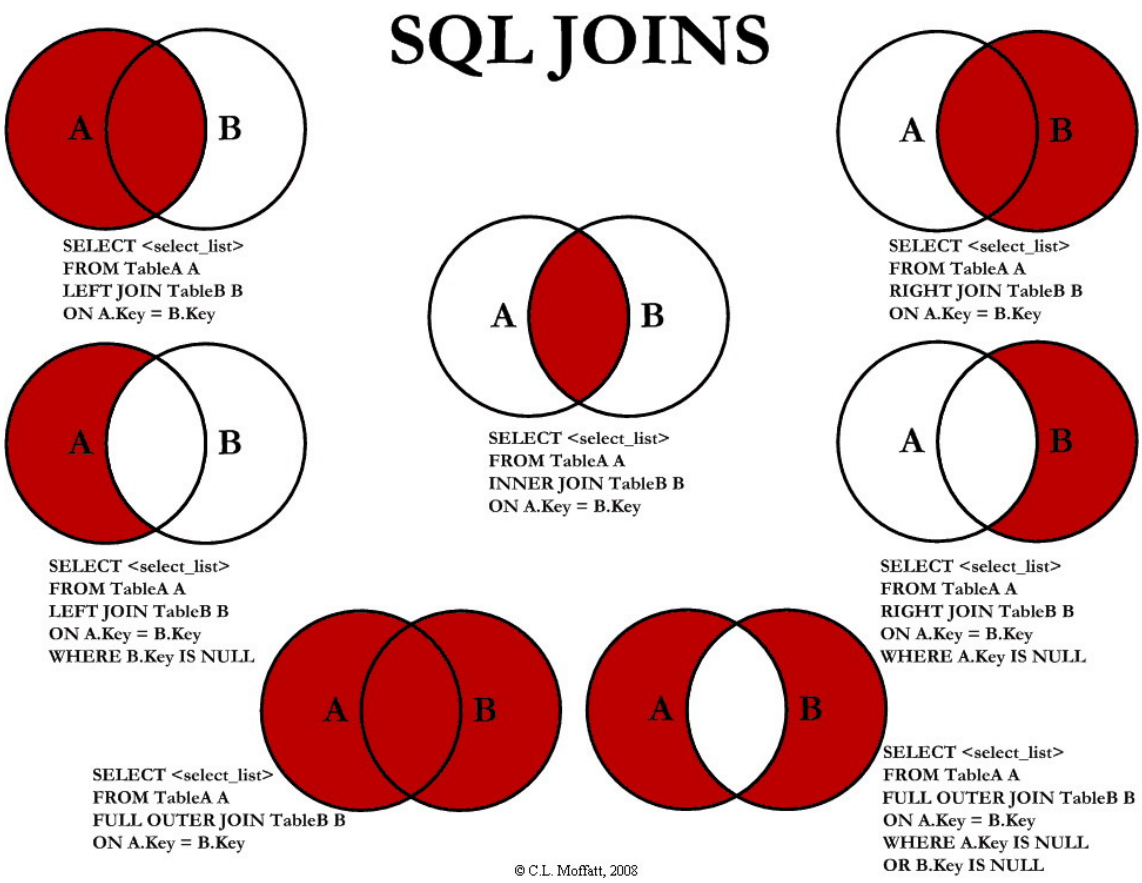
- insert - slouží k vkládání nových záznamů do databáze.

- delete - slouží k mazání záznamů z databáze.

Jazyk pro řízení dat (DCL - Data Control Language) - Slouží pro správu jednotlivých uživatelů a přístupových práv, zprostředkovává podporu pro vkládání, mazaní a úpravu uživatelů a zároveň jejich oprávnění pro úpravy tabulek a pohledů.

Transakční příkazy - transakce představuje ucelenou jednotku SQL příkazů, která se provádí samostatně, díky čemuž zaručují integritu databáze.

Spojení tabulek (SQL příkaz join) - často je potřeba vybírat data podle z více tabulek, k čemuž v SQL slouží příkaz "join". Tabulky lze spojit několika způsoby viz obr. 4



Obrázek 4: SQL Join

2.5 ER diagram (model)

ER diagram je reference na skutečný svět, používají se objekty (entity) a vztahy mezi němi (relace, relationship). Model pracuje s těmito pojmy:

Entita - odpovídá objektům reálného světa (osoba, kniha) a je popsána pomocí svých atributů. Entita musí být jednoznačná oproti jiným entitám a existovat nezávisle na nich, název se uvádí podstatným jménem. Relace - vztah mezi

dvěma a více entitami (vyrábí, půjčuje), určuje kardinalitu mezi entitami, název se uvádí slovesem. Atribut - funkce, která přiřazuje entitám nebo relacím vlastnost (jméno, bydliště). Atributy dělíme na

- Jednoduché - atomická hodnota (jméno, příjmení).
- Složená - skládá se z více atomických hodnot (bydliště - ulice, č. p., město).

Entity dělíme na dva typy:

- Silná entita - entita existuje nezávisle na jiné entitě, entita má svůj primární klíč (primary key).
- Slabá entita - někdy nelze entitu jednoznačně rozlišit pomocí atributů, entita nemá primární klíč, k rozlišení potřebujeme relaci s další entitou (silnou).

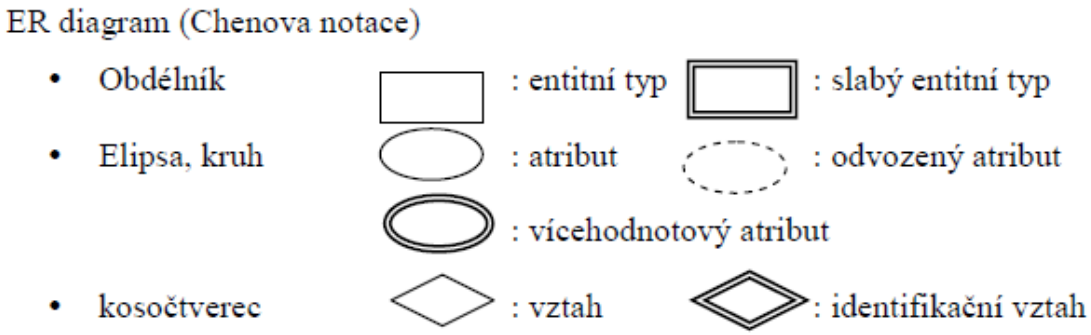
Primární klíč je vybraný kandidátní klíč entitního typu, identifikující každou entitu [1].

Kardinalita - Integritní omezení ER modelu se týkají identifikačních klíčů, speciálně primárního klíče, referenční integrity, domény atributů, kardinality a členství ve vztahu[1]. Kardinalitu binárního vztahu mezi dvěma entitami E1 a E2 dělíme do tří typů:

- vztah 1:1 - jedné E1 odpovídá nejvýše jedna E2 a opačně, jedné E2 odpovídá nejvýše jedna E1. Například osoba - občanský průkaz
- vztah 1:N - jedné E1 odpovídá několik E2, jedné E2 odpovídá pouze jedna E1. Například osoba - Například osoba - trvalé bydliště (na jednom místě může bydlet několik osob, jedna osoba může bydlet pouze na jednom místě)
- vztah N:M jedné E1 odpovídá několik E2, jedné E2 odpovídá několik E1. Osoba - škola (osoba může studovat na více školách, ve škole studuje více osob)

Členství entit v relaci vyjadřuje, zda entita se účastní relace, nebo do vztahu nemusí vstupovat. Podle toho rozlišujeme, povinné a nepovinné členství v relaci. Počet výskytů entity ve vztahu je určen dvojící minimálního a maximálního výskytu. Například čtenář si může vypůjčit žádnou, ale také N knih (0,N). Zatímco kniha může být půjčena maximálně jednomu čtenáři (0,1).

Značení entit, relací, atributů v ER-Diagramu.



Obrázek 5: Značení v ER-diagramu

2.6 Převod ER diagramu do relačního modelu

K převodu použijeme algoritmus uvedený v [1]. Originální algoritmus bude uveden v příloze.

Cílem je převést entity a vztahy do relačního modelu (případně SQL) s minimalizací ztrát. Algoritmus si můžeme rozdělit do šesti částí. Každá část se věnuje jiné kardinalitě. Příklady si budeme ukazovat na entitě Zaměstnanec a entitě Počítač.

- Vztah 1,1 : 1,1 - Každý zaměstnanec má pravě jeden počítač. Každý počítač má pravě jednoho zaměstnance. Obě entity můžeme sloučit do jedné. Vybereme si jednu kterou ponecháme a z druhé vezmene všechny atributy. Primární klíč z druhé entity označíme jako UNIQUE A NOT NULL, abychom zachovali kardinalitu.

- Vztah 1,1 : 0,1 - Každý zaměstnanec má pravě jeden počítač. Počítač má buď jednoho nebo žádného zaměstnance. Takže může být nevyužíván. První entitu doplníme o množinu primárních klíčů z druhé entity, množinu označíme jako UNIQUE (abychom dodrželi kardinalitu). Primární klíč z druhé entity bude navíc cizím klíčem.

- Vztah 0,1 : 0,1 - zaměstnanec má buď jeden nebo žádný počítač. Počítač vlastní jeden nebo žádný zaměstnanec. U této kardinality se již bohužel neobejdeme bez pomocné entity. Která bude obsahovat primární klíče z obou entit. Tyto klíče budou navíc cizí klíče a budou UNIQUE pro zachování kardinality. Původní entity zůstavají nezměněny.

- Vztah 0(1),N : 1,1 - zaměstnanec má libovolný počet počítačů (vztah 0,N) nebo jeden a více počítačů (vztah 1,N). Každý počítač je vlastněn právě jedním zaměstnancem. První entita zůstane nezměněna, k druhé se přidá primární klíč z první entity. Primární klíč již nebude UNIQUE. Primární klíč bude cizí klíč.

- Vztah 0(1),N : 0,1 - zaměstnanec má libovolný počet počítačů (vztah 0,N) nebo jeden a více počítačů (vztah 1,N). Počítač je vlastněn jedním nebo žádným pracovníkem. Opět budeme muset použít pomocnou entitu. Primární klíče z obou entit budou v nové entitě primární klíče, zároveň budou cizí klíče. Tentokrát klíč z první entity nebude UNIQUE. Klíč z druhé entity UNIQUE bude.

- Vztah 0(1, N : 0(1),N - zaměstnanec má libovolný počet počítačů (vztah 0,N) nebo jeden a více počítačů (vztah 1,N). Počítač to má obdobně. Opět budeme používat pomocnou entitu. Vše bude stejné jako o bod výše, pouze klíč z druhé entity nebude UNIQUE.

Terciální vztah. Vztah, který má tři entity. Vztah si rozdělíme na tři podvztahy. Pro každou množinu zvlášť. Tedy E1 - E2, E1 - E3, E2 - E3. Při konverzi se můžeme setkat s těmito problémy, které musíme vyřešit:

- Stejná logická jména sloupců se stejnou (ve vazbách) nebo různou sémantikou. Hlavně platí u vztahu 1,1 : 1,1. Není od věci mít u každého atributu prefix původní entity.

- Pokud mnoho vazeb v ER modelu tvoří sekvence nebo cykly, může dojít k nežádoucím efektům, které v konečném důsledku představují neúplnost, zkreslení, omezení nebo chybu.

ER model	**Relační model**
Entitní typ .Entita	relace
1:1 nebo 1:N typ vztahu	cizí klíč (nebo vztahová relace)
M:N typ vztahu	vztah. relace a dva cizí klíče
n-ární typ vztahu	vztah. relace a n cizích klíčů
jednoduchý atribut	atribut
kompozitní atribut	množina atributů
vícehodnotový atribut	relace a cizí klíč
množina hodnot	Doména
klíčový atribut	Primární, kandidátní klíč

Obrázek 6: Porovnání ER-diagramu a relačního modelu

3 Programátorská příručka

3.1 Použité technologie

Aplikace je naprogramována v jazyce C ♯. Pro grafické uživatelské prostředí byla zvolena technologie Windows Forms, která je součástí frameworku Microsoft .NET Framework, který obsahuje základní grafické funkce. Pro tvorbu aplikace bylo vývojové prostředí Microsft Visual Studio 2013.

3.2 Vykreslení diagramu

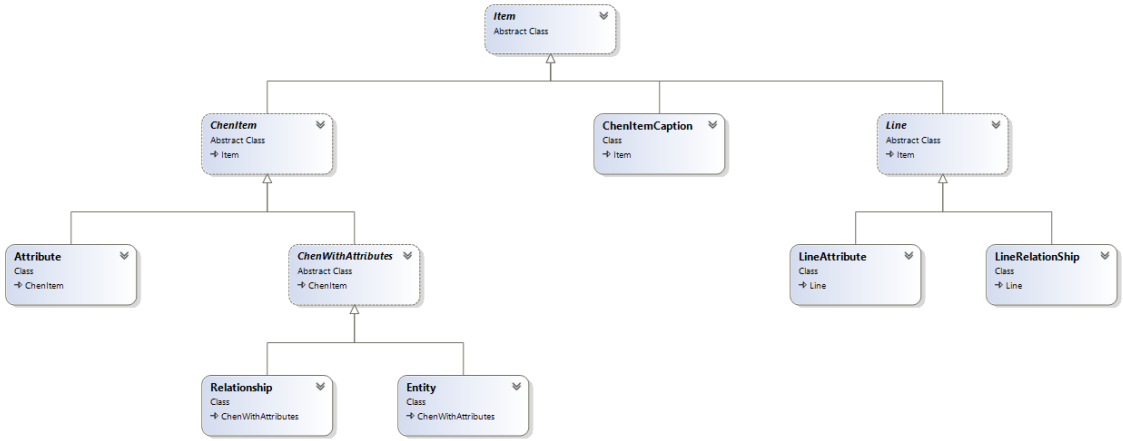
Pro vykreslování diagramu byl navržen vykreslovací engine, který není součástí .NET Frameworku. Engine lze rozdělit na dvě základní části. První část je třída ZoomPicBox (ukázka implementace 1), který vymezuje místo, kde se bude diagram kreslit. Jeho hlavní funkcionalitou je přiblížení/oddálení a možnost použít vertikální a horizontální scrollbar. Ukázka implementace třídy je v příloze. Druhá část je třída DrawSpace, která vykresluje samotný diagram, obsluhuje jednotlivé komponenty diagramu, obsluhuje události jako je klik myši, stisknutí klávesnice, pohyb myší. Stará se o vykreleslení všech tří notací. SQL se vypíše v novém okně.

Prakticky vykreslování funguje následovně. Na hlavní okno formuláře je vykreslen podle požadované velikosti ZoomPicBox, do něj se vykreslí DrawSpace. A konečně do DrawSpace se vykreslují jednotlivé komponenty diagramu.

Popis tříd komponent použitých v diagramu

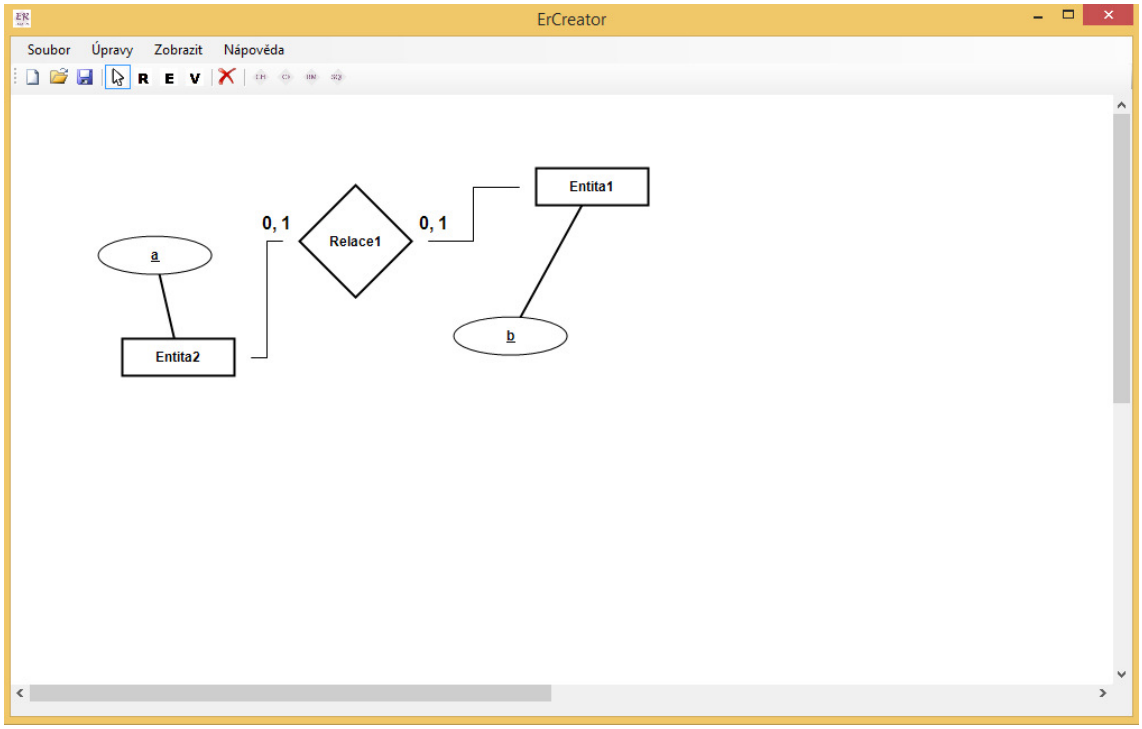
- Item - abstraktní třída, základní předpis pro všechny další komponenty diagramu. Obsahuje události pro klik myši, označení komponenty, vykreslení komponenty, smazaní komponenty, atp.
- ErItem - abstraktní třída, dědí z třídy Item. Předpis pro komponenty Atribut, Entita, Relace.
- ErItemCaption - dědí z třídy Item. Implementace vykreslení názvu komponenty. Popisuje použitý font a text.
- Line - abstraktní třída, dědí z třídy Item. Předpis pro třídy LineAttribute a LineRelationShip
- Attribute - dědí z třídy ErItem. Implementace Attributu, Entity nebo Relace,
- ErWithAttributes - abstraktní třída, dědí z třídy ErItem. Předpis pro komponenty Entita a Relace. Konkrétně implementuje editaci atributů.
- RelationShip - dědí z třídy ErWithAttributes. Třída, které implementuje Relace.
- Entity - dědí z třídy ChenWithAttributes. Třída, které implementuje Entity.

Hiearchive tříd GUI obrázek 7.



Obrázek 7: Hiearchive tříd GUI

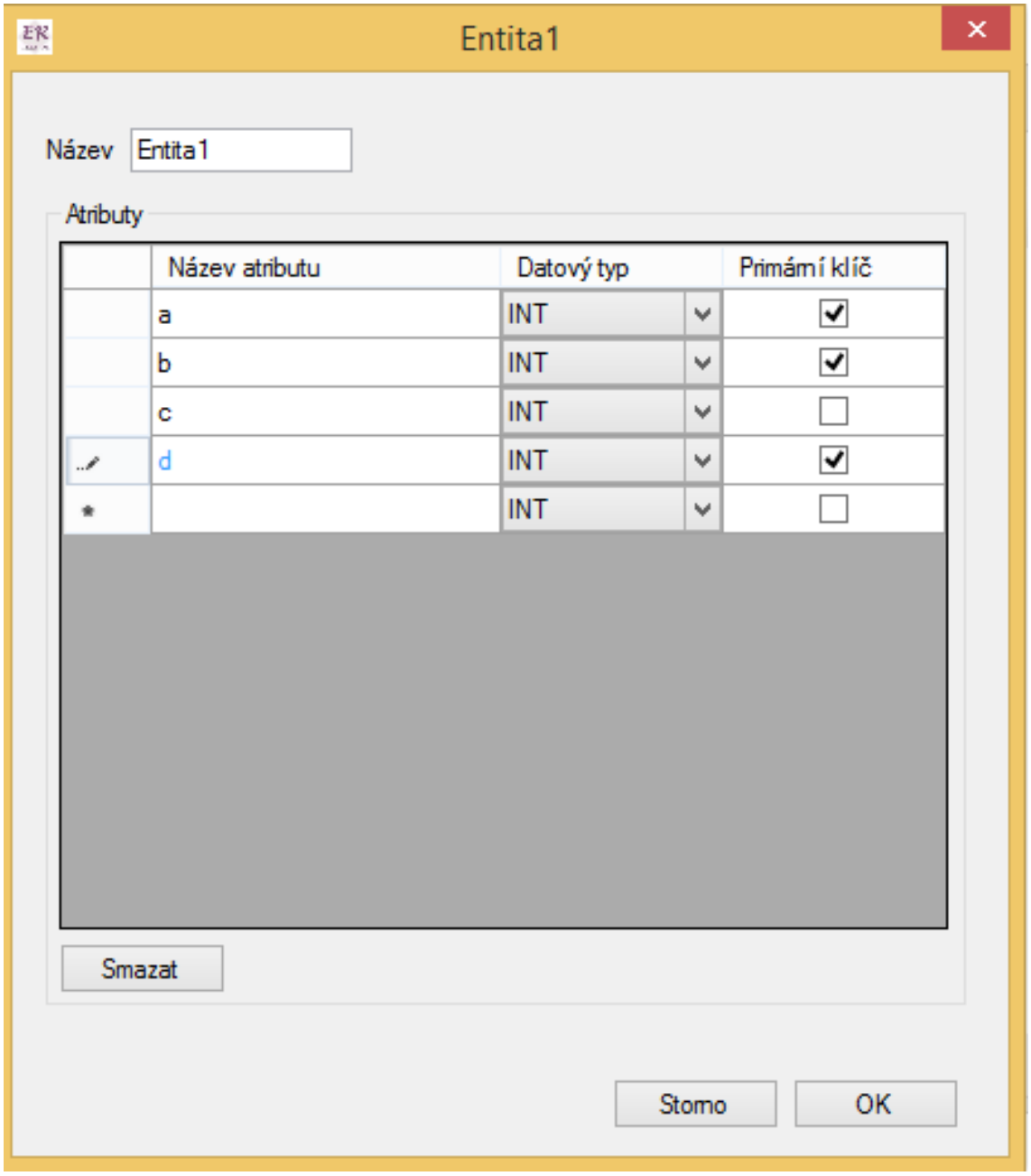
4 Uživatelská příručka



Obrázek 8: Aplikace ErCreator

4.1 Editace entity, relace, atributu

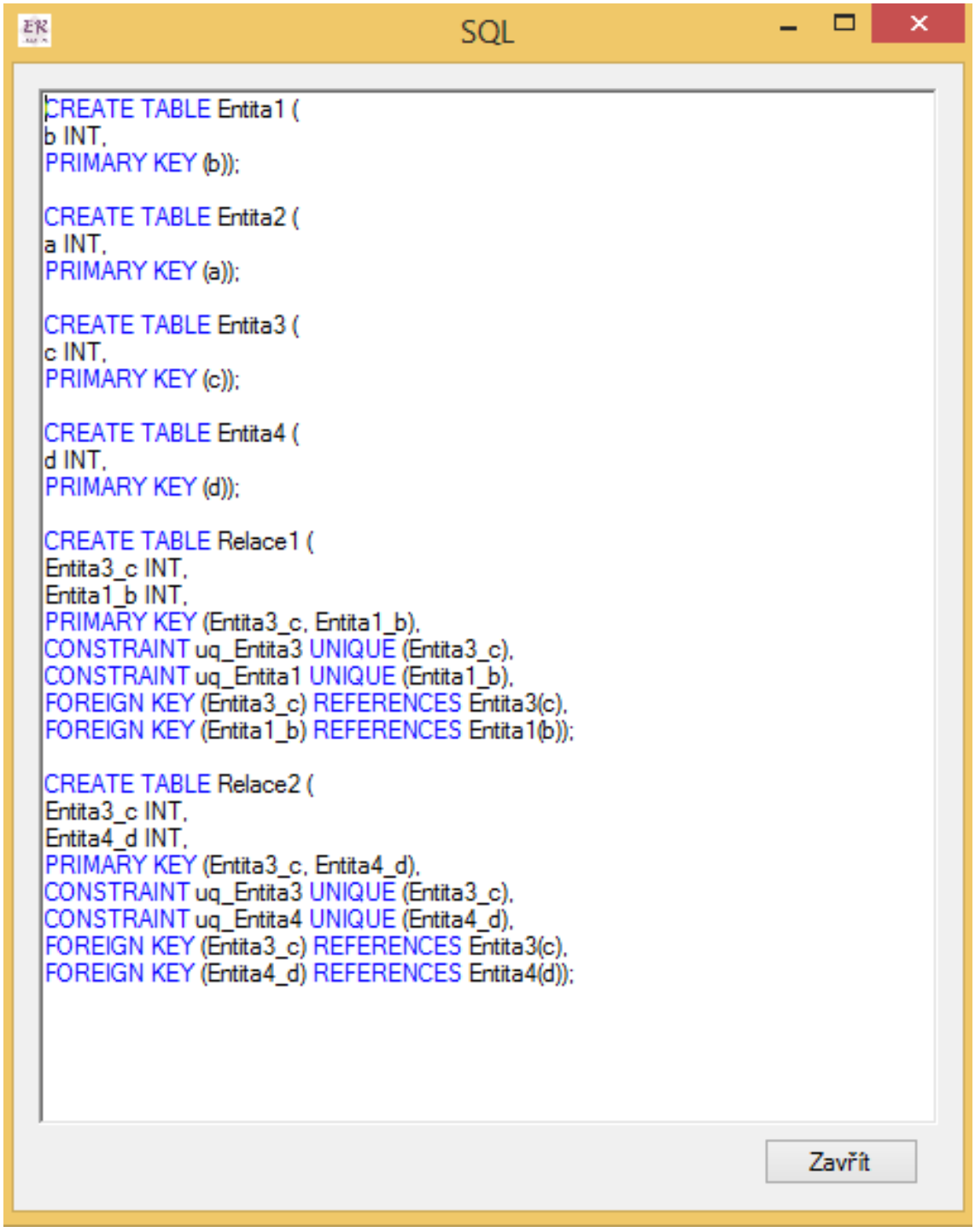
Při dvojkliku na entitu lze měnit její atributy. Typ atributu a primářní klíč. Lze měnit název entity. Při dvojkliku na relaci lze měnit vztahy s entitama. Lze měnit název relace. Při dvojkliku na atribut lze měnit jeho název.



Obrázek 9: Editace entity

4.2 Převod do SQL

Příklad obrázek 10



Obrázek 10: Příklad převodu do SQL

4.3 Podrobný popis aplikace

4.3.1 Menu a toolbar

Nabídka Soubor obsahuje položky:

- Nový - vytvoří nový diagram.
- Otevřít - otevře dialog pro otevření diagramu.
- Uložit jako - zobrazí ukládací dialog
- Uložit jako obrázek - uloží ER právě vybraný typ notace do obrázku
- Zavřít - ukončí aplikaci

Nabídka Úpravy obsahuje položky:

- Vložit relaci
- Vložit entitu
- Vložit vztah mezi relaci a entitu

Nabídka Zobrazení nabízí položky:

- Zobrazit Chenovu notaci
- Zobrazit Crow's foot notaci - zobrazí diagram v této notaci
- Zobrazit Relační model - převede notaci do relačního modelu
- Zobrazit SQL - převede notaci do relačního modelu a zobrazí SQL

4.3.2 Ovládání

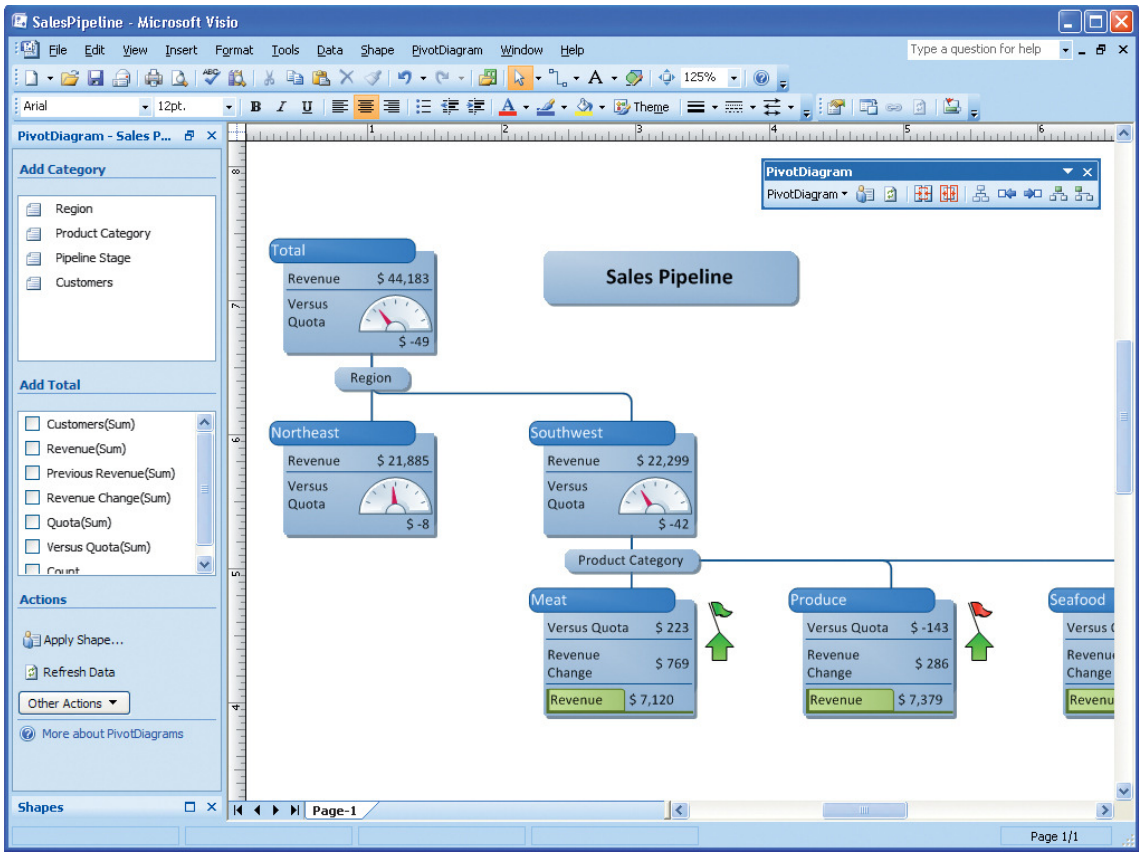
Aplikace se ovládá myší (klikem, dvojklikem), případně chycení pro přesun. Případně lze mazat na klávesnici Tlačítkem Del.

5 Srovnání s podobnými aplikacemi

5.1 Microsoft Visio

Microsoft Visio Standard 2013 je určen pro vizualizaci a tvorbu diagramů, a to buď statických, nebo dynamických, které se automaticky mění v závislosti na datech a v reálném čase. Visio slouží k tvorbě organizačních a síťových diagramů, modelování obchodních procesů nebo schémat podlaží, výrobních linek, ISO procesů a schémat architektury IT. Schémata lze sdílet pomocí webového rozhraní nebo publikovat na server SharePoint.

- **Pozitiva**
  - Aplikace má velmi intuitivní ovládání na plátně.
- **Negativa**
  - Aplikace neumožňuje další práci s modelem. Aplikace slouží pouze pro grafickou ukázku.
  - Aplikace neobsahuje grafickou historii.
  - Aplikace neobsahuje příklady.

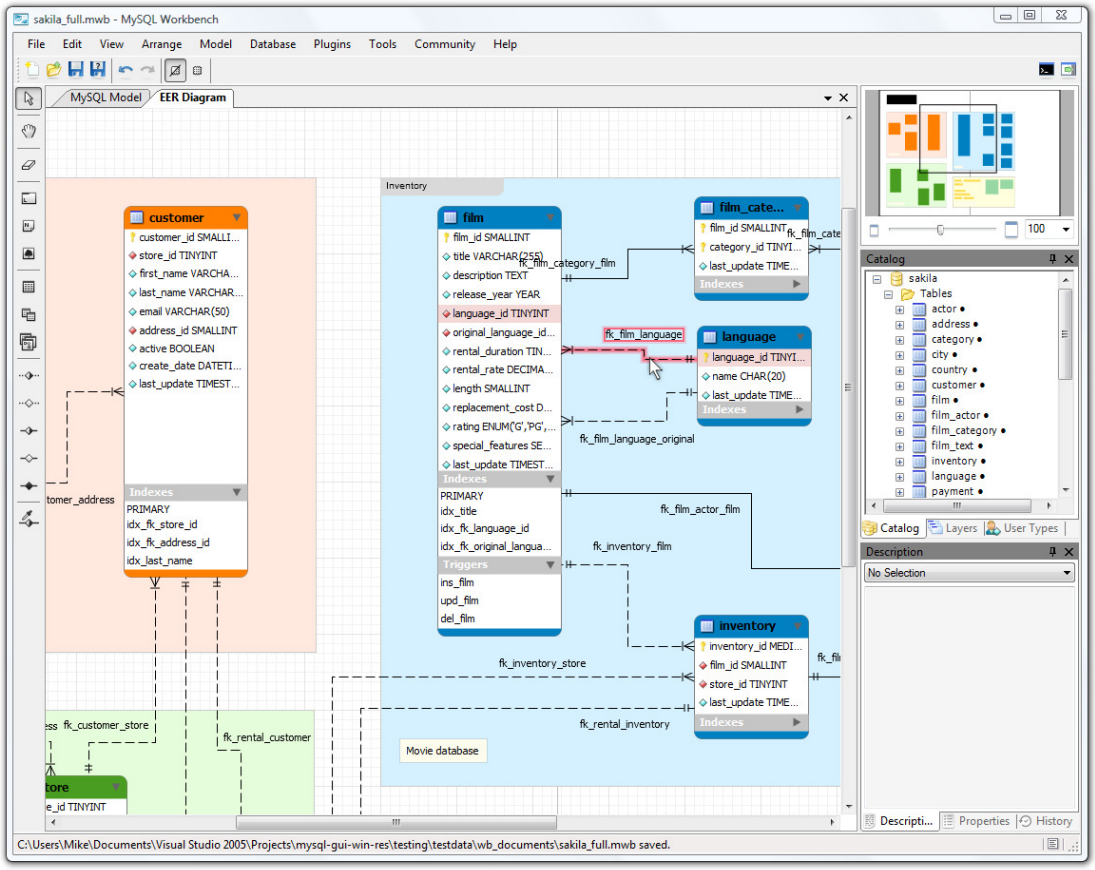


Obrázek 11: Microsoft Visio

5.2 MySQL Workbench

MySQL Workbench je visuální multiplatformní návrhář pro MySQL databáze. Umožňuje vytvářet schémata tabulek spolu s jejich sloupečky a atributy a také relace mezi jednotlivými tabulkami. Zároveň umí administrovat MySQL server, čímž nahrazuje aplikaci MySQL Administrator a také lze použít pro vývoj, čímž nahrazuje MySQL Query Browser.

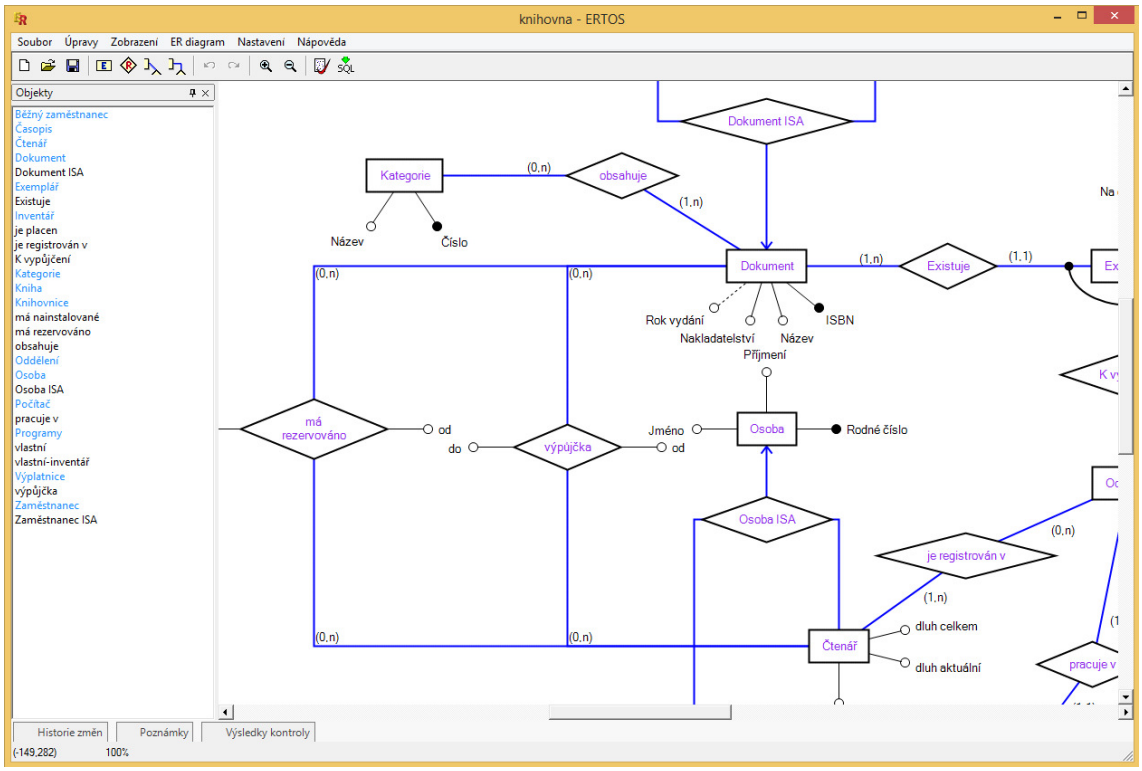
Ve srovnání s aplikací ErCreator používá aplikace Crow's feet notacinotaci pro vytváření diagramů, umožňuje dále převod do SQL. Neumožňuje převod do jiných notací



Obrázek 12: MySQL Workbench

5.3 Ertos

Aplikace, která je nejvíce podobná aplikace ErCreator. Při vytváření jsem hodně z této aplikace vycházel. Ertos neumožňuje převod do relačního modelu ani nepodporuje další notace.



Obrázek 13: Ertos

```
\label{code:zoom}
class ZoomPicBox : ScrollableControl
   {

      private void UpdateScaleFactor()
      {
         if (_image == null)
            this.AutoScrollMinSize = this.Size;
         else
         {
            this.AutoScrollMinSize = new Size(
              (int)(this._image.Width * _zoom + 0.5f),
              (int)(this._image.Height * _zoom + 0.5f)
              );
         }
      }

      protected override void OnPaint(PaintEventArgs e)
      {
         //if no image, don't bother
         if (_image == null)
         {
            base.OnPaintBackground(e);
            return;
         }
         //Set up a zoom matrix
         Matrix mx = new Matrix(_zoom, 0, 0, _zoom, 0, 0);
         //now translate the matrix into position for the scrollbars
         mx.Translate(this.AutoScrollPosition.X / _zoom, this.
            AutoScrollPosition.Y / _zoom);
         //use the transform
         e.Graphics.Transform = mx;
         //and the desired interpolation mode
         e.Graphics.InterpolationMode = _interpolationMode;
         //Draw the image ignoring the images resolution settings.
         e.Graphics.DrawImage(_image, new Rectangle(0, 0, this.
            _image.Width, this._image.Height), 0, 0, _image.Width,
            _image.Height, GraphicsUnit.Pixel);
         base.OnPaint(e);
      }

      protected override void OnMouseWheel(MouseEventArgs e)
      {
         int numberLinesToMove = e.Delta * SystemInformation.
            MouseWheelScrollLines / 120;
         Zoom += numberLinesToMove * 0.01f;
      }
   }
```

Zdrojový kód 1: C#

Závěr

Cílem práce bylo vytvořit aplikaci, která bude umožňovat pohodlné vytváření ER modelu, převod do SQL, převod do relačního modelu. Byl kladen důraz na jednoduché a intuitivní ovládání. Zlepšení aplikace by mohlo spočívat v možné editaci relačního modelu, širší editaci v Crow's feet modelu.

A Originální algoritmus pro převod ER modelu do relačního modelu

Text první přílohy

Cílem je převést entity a vztahy na předpis záhlaví pro tabulky v relačním modelu a zároveň minimalizovat ztráty s přechodem do nižší úroveň abstrakce. Budeme tedy ke schématu připojovat integritní omezení.

- Vytvoření relací z regulárních (silných) entitních typů, atributy relací odpovídají jednoduchým atributům entitních typů, u složených typů atributů provedeme dekompozici na jednoduché atributy. Pokud je struktura složitější, transformujeme příslušné subschéma ER diagramu do pro transformaci použitelné podoby. Převezmeme, primární klíč.

  Relace: Dokumentace(IDd(PK), Projekt)

- K transformaci slabého entitního typu potřebujeme znát relaci identifikačního vlastníka. Provedeme transformaci jako v předešlém případě. Dostaneme pouze parciální klíč. Doplníme relaci o atributy identifikačního klíče relace identifikačního vlastníka (tvoří cizí klíč) a přidáním k parciálnímu klíči definujeme primární klíč.

- Typu vztahu odpovídá schéma relace, zahrnující referenční integritní omezení. Obecně pro všechny transformace existuje více variant a záleží na dalších vazbách a IO (povinné a nepovinné členství ve vztahu), předpokládaných datech (rozsah, počet (relativní) propojených entit) a převaţujících dotazech. Problematiku naznačíme na příkladech.

  - Při vztahu (1,1)-(1,1), každý pracovník má právě jeden počítač se nabízí možnost spojit entitní typy do jedné relace, zvolit primární klíč z jednoho entitního typu, sloupec původního primárního klíče druhé relace (idP) doplníme o IO UNIQUE. Sémanticky z pracovního počítače uděláme charakteristiku zaměstnance. Toto řešení nepředpokládá další vazby na entitní typ počítač.
  - Jinak transformaci provedeme převedením na dvě relace. Do záhlaví jedné (Zaměstnanec) přidáme jako vazební položku primární klíč druhé (idP), připojíme IO – referenční integritu, UNIQUE, případně NOT NULL. Nový sloupec je cizím klíčem, NOT NULL kontroluje povinné členství ve vztahu. Tedy pokud by počítač neměl každý zaměstnanec – vztah (0,1)(1,1), potom NOT NULL nepoužijeme.
  - Pro některé případy (velká záhlaví) a převažující užívání speciálních dotazů (např. na atributy vazby) může být výhodné provést transformaci do tří relací. Dvě entitní relace jsou definovány entitními typy a záhlaví třetí vztahové relace (pro typ vztahu) vytvoříme z primárních klíčů ve vazbě zúčastněných entitních typů. Ty jsou opět cizími klíči,

UNIQUE a NOT NULL. V kombinaci potom tvoří primární klíč. Toto řešení je sémanticky srozumitelné třeba v situaci, kdy vztahový typ má další atributy.

- Vztah (1,N)(1,1) – všechny místnosti jsou obsazeny, všichni zaměstnanci jsou umístěni řešíme nejčastěji vytvořením dvou relací. Do záhlaví relace(Zaměstnanec), která je ve vazbě s kardinalitou 1 přidáme primární klíč z entitního typu(Místnost), který ve vazbě reprezentuje kardinalitu N. Ten je cizím klíčem s dalším integritním omezením NOT NULL (povinné členství ve vztahu), ale již bez UNIQUE.
- Podobně řešíme reprezentace vztahů N:M. Pokud každý zaměstnanec pracuje nejméně na jednom úkolu a každý úkol je řešen nejméně jedním zaměstnancem, vztah lze popsat jako (1,N) (1,M).
- Zvláštními případy, které se ale řeší analogicky jsou unární (rekurzívní) vazby. Např. N : 1. Do relace Zaměstnanec přidáme opět vazební sloupec (IDvedoucí), jehož doména je identická s IDz entitního typu Zaměstnanec.
- Naopak n-ární vztah při povinném členství pro různé kombinace kardinalit transformujeme tak, že definujeme jednu vztahovou relaci se záhlavím složeným z primárních klíčů všech v relaci zúčastněných entitních typů.

Při konverzi se můžeme setkat s těmito problémy, které musíme vyřešit:

- Stejná logická jména sloupců se stejnou (ve vazbách) nebo různou sémantikou. Proto používáme výstižné, úplné unikátní pojmenování, případně tečkové notace.
- Pokud mnoho vazeb v ER modelu tvoří sekvence nebo cykly, může dojít k nežádoucím efektům, které v konečném důsledku představují neúplnost, zkreslení, omezení nebo chybu.

B Obsah přiloženého CD/DVD

Na samotném konci textu práce je uveden stručný popis obsahu přiloženého CD/DVD, tj. jeho závazné adresářové struktury, důležitých souborů apod.

bin/
: Instalátor Instalator programu, popř. program Program, spustitelné přímo z CD/DVD. / Kompletní adresářová struktura webové aplikace Webovka (v ZIP archivu) pro zkopírování na webový server. Adresář obsahuje i všechny runtime knihovny a další soubory potřebné pro bezproblémový běh instalátoru a programu z CD/DVD / pro bezproblémový provoz webové aplikace na webovém serveru.

doc/
: Text práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech příloh, a všechny soubory potřebné pro bezproblémové vygenerování PDF dokumentu textu (v ZIP archivu), tj. zdrojový text textu, vložené obrázky, apod.

src/
: Kompletní zdrojové texty programu Program / webové aplikace Webovka se všemi potřebnými (příp. převzatými) zdrojovými texty, knihovnami a dalšími soubory potřebnými pro bezproblémové vytvoření spustitelných verzí programu / adresářové struktury pro zkopírování na webový server.

readme.txt
: Instrukce pro instalaci a spuštění programu Program, včetně všech požadavků pro jeho bezproblémový provoz. / Instrukce pro nasazení webové aplikace Webovka na webový server, včetně všech požadavků pro její bezproblémový provoz, a webová adresa, na které je aplikace nasazena pro účel testování při tvorbě posudků práce a pro účel obhajoby práce.

U veškerých cizích převzatých materiálů obsažených na CD/DVD jejich zahrnutí dovolují podmínky pro jejich šíření nebo přiložený souhlas držitele copyrightu. Pro všechny použité (a citované) materiály, u kterých toto není splněno a nejsou tak obsaženy na CD/DVD, je uveden jejich zdroj (např. webová adresa) v bibliografii nebo textu práce nebo v souboru `readme.txt`.

Literatura

[1] HRONEK, Jiří. *Databázové systémy*. 2007.

[2] VYCHODIL, Vilém. *Databázové systémy*. 2013.

[3] SHARP, John. *Microsoft Visual C ♯ 2010 krok za krokem*. 2010.