PŘÍRODOVĚDECKÁ FAKULTA UNIVERZITY PALACKÉHO
KATEDRA INFORMATIKY

DIPLOMOVÁ PRÁCE

Dotazovací jazyk pro relační podobnostní databáze

2014 Bc. Ondřej Vaverka

Anotace

Práce se věnuje logickým modelům dat, jež podporují podobnostní dotazy. Práce nabízí přehled hlavních přístupů, podrobněji pak popisuje zobecnění relačního modelu založené na fuzzy logice v úzkém slova smyslu. Jelikož ve fuzzy logice obecně není možné vzájemně definovat kvantifikátory, musí být v zobecněné relační algebře zahrnuta i operace relačního dělení. Práce poskytuje přehled různých definic relačního dělení včetně jejich zobecnění a ukazuje, že v praktickém jazyce je možné při zachování relační úplnosti namísto dělení uvažovat „image relations“, jejichž použití je pro uživatele přirozenější. Dále se práce věnuje operacím GROUP a UNGROUP, jejichž zobecněné varianty umožňují implementaci zobecněného modelu pomocí klasického relačního databázového systému.

Děkuji doc. RNDr. Vilému Vychodilovi, Ph. D. za cenné rady, připomínky, ochotu a velkou trpělivost. Děkuji Martině Zellerové za psychickou a fyzioterapeutickou pomoc.

Obsah

Seznam tabulek

1. Úvod

Cílem práce je prozkoumat možnosti rozšíření logického relačního modelu dat podporujícího podobnostní dotazy. Práce se věnuje operacím relačního dělení, které musejí být v některé své podobě v modelu zahrnuty jako základní operace. Na rozdíl od klasického modelu nejsou tyto operace odvoditelné a jejich přítomnost je nutná pro vyjádření dotazů se všeobecným kvantifikátorem a pro prokázání úplnosti relační algebry vzhledem k relačnímu kalkulu. Dále je pozornost věnována operacím `GROUP` a `UNGROUP`, které lze mimo jiné využít pro přehlednější reprezentaci výsledku dotazu. Jejich zobecněná varianta však hraje další důležitou roli, kde tyto operace v jistém smyslu vytváří most mezi klasickým a zobecněným modelem. Přesněji řečeno, umožňují chápat zobecněné relace s ranky jako klasické relace, které jsou rozšířeny o atribut *rank*.

Proč se vůbec zabývat podporou podobnostních dotazů a jaký přístup zvolit? Nedílnou součástí databázového systému by měla být schopnost zodpovídat dotazy nad daty, které jsou v tomto systému uloženy. V opačném případě nejsou uložená data k užitku. Z historického pohledu, před příchodem relačního databázového modelu, bylo dotazování velmi komplikované a vyžadovalo složité programy i pro jednoduché dotazy. Příkladem může být síťový (`CODASYL`) model, kdy bylo pro získání dat nutné procházet graf, ve kterém byly vazby mezi daty zachyceny pomocí ukazatelů. [24]

O zásadní změnu se zasloužil Edgar F. Codd svým relačním databázovým modelem, který představil poprvé v zásadním článku *A relational model of data for large shared data banks* [11]. Tento článek začíná slovy, že budoucí uživatelé databázových systémů by měli být uchráněni před nutností znát konkrétní (fyzický) způsob uložení dat v těchto systémech. Z pohledu relačního modelu jsou data i jejich vzájemné vztahy reprezentovány pomocí relací. Výsledky dotazů jsou opět relacemi. Pro manipulaci s relacemi zavedl Codd v tomto článku několik operací, které později nazval jako relační algebra. V následujících článcích [12] [13] Codd definoval relační kalkul a dokázal, že relační algebra a relační kalkul jsou z hlediska dotazování rovnocenné. Z relační algebry a relačního kalkulu se později vyvinuly dotazovací jazyky jako `SQL` nebo `QUEL`.

Soudobé databázové systémy (do určité míry) vycházejí z relačního databázového modelu, disponují vysokoúrovňovým dotazovacím jazykem, odstiňují uživatele od fyzické implementace a podporují efektivní vyhodnocování dotazů nad daty. Většina systémů však dotazy vyhodnocuje z pohledu klasické dvouhodnotové logiky. Daný záznam buď dotazu plně vyhovuje a je zahrnut do výsledku, anebo dotazu nevyhovuje. Tento fakt však může pro uživatele představovat několik problémů.

Pokud uživatel data uložená v databázi alespoň rámcově nezná, může se stát, že jeho dotaz bude příliš restriktivní. Dotazu nebude vyhovovat žádný záznam a databázový systém vrátí prázdný výsledek. Nemusí být zřejmé, kterou část

dotazu je nutné upravit, případně jakým způsobem. Navíc se může stát, že po úpravě bude dotazu vyhovovat příliš mnoho záznamů a uživatel bude muset svůj dotaz dále upravovat.

Uživatel musí svůj dotaz vyjádřit přesně, s čímž se pojí další možné problémy. Může se stát, že objekt, který by uživateli vyhovoval nejvíce, leží těsně za hranicemi dotazu. Jako příklad lze uvést databázi autobazaru nebo realitní kanceláře. Uživatel svou představu (např. automobil v cenové relaci kolem 100 000 Kč) musí převést do přesného dotazu (cenové rozmezí od 80 000 Kč do 120 000 Kč). Automobil, který by se uživateli líbil nejvíce, však stojí 125 000 Kč a ve výsledku dotazu proto není zahrnut. Rozdíl 5 000 Kč z hlediska lidského vnímání hraje pramalou roli, z pohledu databázového systému jde o rozdíl významný. Tento efekt je dále umocněn, pokud dotaz zahrnuje větší množství takových podmínek.

Vyjádření představ uživatele pomocí přesného dotazu nemusí být vždy intuitivní. U numerických hodnot (cena, počet pokojů, ...) je uvažování v pojmech od-do ještě relativně přirozené. Nicméně v případě kvalitativních hodnot (žánr, typ oboru, ...) již podobný přístup není možný. Jeden z možných postupů je následující. Nejprve je nutné zjistit všechny podobné hodnoty a poté pomocí dílčích dotazů sestavit výsledek původně zamýšleného dotazu. Některé podobné hodnoty mohou být již svým významem původní hodnotě vzdálené a dílčí výsledky odpovídající těmto hodnotám tak pro uživatele méně relevantní. V celkovém výsledku však tyto méně relevantní odpovědi nemusí být od relevantních odlišeny, a tím dále ztíží uživateli práci s databází.

Databázové systémy by měly umožnit zadávání podobnostních dotazů a výsledky by měly být uživateli předkládány seřazené podle relevance. Navíc by se databázové systémy měly zaměřit na efektivní vyhodnocování těchto dotazů. Protože uživatele většinou méně relevantní výsledky nezajímají, databázové systémy by měly být schopny počítat výsledky postupně počínaje od nejvíce relevantních.

Přístupů k řešení tohoto problému existuje celá řada. Přístupy vycházejí z relačního modelu a různým způsobem jej rozšiřují. Většina přístupů se však nezabývá vztahem rozšíření k logickým základům relačního modelu. Právě úzký vztah relačního modelu k predikátové logice se považuje za hlavní důvod úspěchu relačního modelu [21]. Díky tomuto vztahu bylo možné model dobře prozkoumat a efektivně implementovat. Na této myšlence bylo navrženo zobecnění Coddova relačního modelu pomocí fuzzy logiky v úzkém slova smyslu. Namísto různých rozšíření se uvažuje zobecnění relačního modelu [3] pomocí obecnější struktury pravdivostních hodnot. Koncept podobnosti se stává přirozenou součástí modelu. Navíc díky zachování úzkého vztahu k logice lze dobře zkoumat další pojmy, které s modelem souvisejí – závislosti v datech, integritní omezení, apod.

2. Přehled přístupů

Většina přístupů se snaží poskytnout uživateli větší volnost při zadávání dotazu. Umožňují pokládání podobnostních, „tolerantních“, či preferenčních dotazů. Některé přístupy navíc řadí výsledky podle relevance. Podle hlavních rysů lze přístupy rozdělit do několika kategorií.

Přístupy využívající ranky, bez využití fuzzy logiky

Při vyhodnocování dotazu je záznamům přiřazena numerická hodnota – rank (skóre). Význam ranku je intuitivní – čím vyšší rank daný záznam má, tím lépe odpovídá dotazu a tím více je pro uživatele relevantní. Pro tyto přístupy je typická podpora `Top-K` dotazů, kdy odpověď na dotaz tvoří jen několik prvních nejvíce relevantních záznamů. Mezi zástupce lze zařadit např. `RankSQL` [27].

Přístupy založené na fuzzy logice v širokém slova smyslu

Výsledkem dotazu je fuzzy relace. Stupeň příslušnosti záznamu do výsledné relace je vypočítán na základě stupňů příslušnosti daného záznamu do fuzzy relací v dotazu a splnění podmínek, které se označují jako fuzzy predikáty [36]. Lze se setkat s jinými interpretacemi, které namísto stupně příslušnosti uvažují úroveň závislosti nebo stupeň důležitosti [33]. Dotazy se často označují jako preferenční nebo flexibilní, databázové systémy podporující takové dotazy potom jako fuzzy databáze. Jako zástupce lze uvést `SQLf` [36].

V literatuře se občas pod pojmem „fuzzy databáze“ rozumí rozšíření relačního modelu o možnost použití fuzzy množin namísto přesných hodnot, kdy v případě nejisté nebo vágní hodnoty lze tuto hodnotu modelovat pomocí fuzzy množiny. Tento přístup se také označuje jako posibilistický model.

Přesněji řečeno se o rozšíření nejedná – relační model (v moderním pojetí [17], [20]) nezakazuje použití libovolné domény. Fuzzy množiny lze bez omezení používat i v klasickém relačním modelu. Navíc používání pojmu „fuzzy databáze“ v různých významech (podobnostní dotazování oproti modelování neurčitosti v datech pomocí fuzzy množin) vede ke znepřehlednění této oblasti a horší přístupnosti pro databázovou komunitu [36].

Přístup založený na fuzzy logice v úzkém slova smyslu

V klasickém relačním modelu lze dotaz vyjádřit pomocí formule predikátové logiky prvního řádu [13]. Zjednodušeně řečeno, databáze slouží jako struktura pro jazyk formulí a jednotlivé záznamy z databáze slouží jako ohodnocení volných proměnných této formule. V případě, že je formule pod takovým ohodnocením pravdivá (její pravdivostní hodnota je 1), můžeme říct, že daný záznam formuli splňuje a můžeme jej zařadit do výsledku.

Množinu výsledků lze popsat pomocí její charakteristické funkce. Pro prvky, které do množiny patří a tedy splňují dotaz (formuli), funkce nabývá hodnoty 1, pro ostatní prvky nabývá hodnotu 0. Pro daný záznam je tedy mezi pravdivostní hodnotou formule a charakteristickou funkcí vzájemně jednoznačný vztah.

Klasický relační model je založen na klasické dvouhodnotové logice, jejíž strukturou pravdivostních hodnot je dvouprvková Booleova algebra. Formule může být buď zcela pravdivá, či zcela nepravdivá.

Přístup založený na fuzzy logice v úzkém slova smyslu [3] nahrazuje Booleovu algebru obecnější strukturou pravdivostních hodnot (úplný reziduovaný svaz), která umožňuje uvažovat i mezilehlé stupně pravdivosti. Množina výsledků je nyní fuzzy množina, ve které stupeň příslušnosti odpovídá stupni, ve kterém daný záznam vyhovuje dotazu (formuli).

V klasickém relačním modelu je každá doména vybavena relací identity, ačkoliv se tato relace běžně neuvažuje a implicitně se předpokládá. Identitu lze chápat jako speciální případ relace podobnosti. Dva prvky jsou si buď zcela podobné (jsou stejné), anebo si nejsou vůbec podobné (jsou různé). V přístupu založeném na fuzzy logice v úzkém slova smyslu lze relace podobnosti uvažovat obecně ve stupních.

Na rozdíl od předchozích přístupů tento přístup není rozšířením relačního modelu, ale jeho zobecněním. Tímto přístupem se práce dále zabývá.

Ostatní přístupy

- Lingvistické preference [37]

  Uživatel pomocí klíčových slov `ideal`, `good`, `acceptable` aj. určí své preference. Při vyhodnocování dotazu se nejprve získají záznamy, které víceméně odpovídají zadanému dotazu a poté jsou seřazeny podle zadaných preferencí.

- Přístupy založené na Paretově uspořádání [4] / *skyline* dotazy [36]

  Daný záznam je zahrnut do výsledku, pokud neexistuje jiný záznam, který by nad ním dominoval. Uživatel v dotazu vyjádří své preference. Řekneme, že záznam t dominuje nad t', pokud t splňuje všechny preference alespoň tak dobře jako t' a navíc existuje nějaká preference, kterou t splňuje lépe než t'.

- CP-sítě [8]

  CP-sítě jsou grafickou reprezentací preferencí typu *ceteris paribus*. CP-sítě jsou postaveny na myšlence, že uživatel vyjadřující své preference vždy přemýšlí v nějakém kontextu. Jeho preference určitého atributu předpokládá, že ostatní atributy u uvažovaných objektů zůstávají přibližně stejné. Například preference uživatele „Kulaté stoly jsou lepší než hranaté“ znamená, že pokud by si měl uživatel vybrat ze dvou velmi podobných stolů, z nichž jeden je kulatý a druhý hranatý, bude preferovat kulatý. Neznamená to však, že uživatel preferuje kulatý stůl za každých okolností.

Přístupy se dále liší podle úrovně implementace, která sahá od nativní podpory v databázovém systému (`RankSQL`) až po různé pluginy, které pouze využívají služeb existujících databázových systémů (`SQLf`).

Poznámka Výše uvedené přístupy se zabývají podporou podobnostních dotazů – zdrojem neurčitosti nejsou uložená data, ale vágnost lidského jazyka a způsob lidského uvažování. Tyto přístupy pracují nad přesnými daty stejně jako klasické databázové systémy. K modelování neurčitosti v datech slouží zcela odlišné přístupy, jako jsou například pravděpodobnostní databázové systémy [15] [28]. Příkladem neurčitých dat mohou být měření z vědeckých experimentů nebo výsledky párování dat z různých zdrojů.

2.1. Přístupy využívající ranky – RankSQL

Principem těchto přístupů je při dotazování umožnit ohodnocení záznamů numerickou hodnotou – rankem. Uživatel své požadavky formalizuje pomocí ohodnocovacích funkcí. Pro každý záznam je vypočtena hodnota každé ohodnocovací funkce, následně se z těchto hodnot monotonní agregační funkcí určí výsledný rank záznamu. Záznamy jsou následně podle ranku sestupně uspořádány a v tomto pořadí předkládány uživateli.

Tento typ dotazu je možné provádět i v soudobých relačních databázových systémech. Nejprve bychom provedli selekci záznamů, které nás zajímají. Poté bychom vypočítali ranky všech záznamů, které splnily podmínky selekce. Dále bychom tyto záznamy uspořádali a nakonec uživateli vrátili několik prvních záznamů. Problémem tohoto postupu je jeho neefektivita. Vstupní relace mohou obsahovat velké množství záznamů, výpočet ohodnocovacích funkcí může být drahý a na závěr je nutné všechny záznamy uspořádat – přitom uživatele zajímá jen několik prvních záznamů.

Pro efektivní implementaci je nutné, aby databázový systém podporoval následující optimalizace [27]. Aby nebylo nutné vyhodnocovat rank monoliticky, musí být databázový systém schopen výpočet ranku *rozdělit do více kroků* a umožnit jeho *přeskupení* s ostatními operacemi. Vzhledem k tomu, že ohodnocovací funkce mohou být různě výpočetně náročné, může být vhodné naplánovat výpočet nejméně náročných funkcí zkraje vyhodnocování dotazu a výpočet nejdražších funkcí ponechat až do pozdní fáze vyhodnocení. Díky přeskupení proběhnou před výpočtem nejdražších funkcí selekce, které mohou omezit velikost mezivýsledku. Navíc díky *inkrementálnímu výpočtu* se v ideálním případě budou nejnáročnější ohodnocovací funkce počítat jen pro několik málo záznamů.

Cílem `RankSQL` [27] je zajistit efektivní podporu pro ranky v relačním databázovém systému. Pro tento účel autoři definují rozšíření relační algebry, ve kterém zavádí operaci výpočtu ranku jako základní součást algebry a poskytují algebraické identity nezbytné pro optimalizace (rozdělení výpočtu ranku a přeskupení s ostatními operátory).

2.1.1. Relace s ranky

Základem rozšířené relační algebry jsou relace s ranky [27]. Od klasických relací se liší ve dvou aspektech – záznamy (přesněji n-tice) jsou ohodnoceny numerickou hodnotou (rankem) a dále jsou podle této hodnoty sestupně uspořádány. Rank záznamu t je vypočten pomocí monotonní agregační funkce $\mathcal{F}$ aplikované na hodnoty jednotlivých ohodnocovacích funkcí $p_1, \ldots, p_n$ pro daný záznam.

Protože během zpracování dotazu může dojít k rozdělení a přeskupení operací výpočtu ranku, nemusejí být v daném kroku výpočtu známy hodnoty všech ohodnocovacích funkcí. Pro výpočet ranku záznamů se proto neznámé hodnoty nahrazují maximální možnou hodnotou, kterou může daná ohodnocovací funkce nabývat. Díky monotonii agregační funkce takto vypočtený (dílčí) rank (ozn. $\overline{\mathcal{F}}_{\mathcal{P}}[t]$, kde $\mathcal{P} = \{p_i | \text{hodnota funkce } p_i \text{ je známá}\}$) odpovídá hornímu odhadu výsledného ranku $\mathcal{F}(p_1, \ldots, p_n)[t]$.

Z monotonie agregační funkce plyne následující fakt (*ranking principle* [27]). Platí-li pro dva záznamy t_1 a t_2 nerovnost $\overline{\mathcal{F}}_{\mathcal{P}}[t_1] > \overline{\mathcal{F}}_{\mathcal{P}}[t_2]$, potom, je-li nutné při vyhodnocování dotazu zpracovat záznam t_2, musíme nutně zpracovat i t_1. Z tohoto důvodu je vhodné, aby se v každém kroku výpočtu přednostně zpracovávaly záznamy s nejvyšším dílčím rankem. Použitím relací s ranky společně s inkrementálním výpočtem je toto preferované pořadí zpracování záznamů zajištěno.

Formální definice relace s ranky je následující [27]. Nechť je R relace, $\mathcal{F}$ monotonní agregační funkce a $\mathcal{P} \subseteq \{p_1, \ldots, p_n\}$ množina již vyhodnocených ohodnocovacích funkcí. Potom relace s ranky $R_{\mathcal{P}}$ je relace R rozšířená o:

- speciální atribut `Skóre` – hodnota tohoto atributu pro záznam t je rovna hornímu odhadu ranku tohoto záznamu $\overline{\mathcal{F}}_{\mathcal{P}}[t]$,
- relaci uspořádání $<_{R_{\mathcal{P}}}$ – kde $\forall t_1, t_2 \in R_{\mathcal{P}} : t_1 <_{R_{\mathcal{P}}} t_2 \Leftrightarrow \overline{\mathcal{F}}_{\mathcal{P}}[t_1] < \overline{\mathcal{F}}_{\mathcal{P}}[t_2]$.

Mají-li některé záznamy stejné skóre, jejich vzájemné pořadí je možné rozhodnout například podle unikátních ID, případně jiným deterministickým způsobem.

2.1.2. Operace rank-relační algebry

Operace rank-relační algebry sestávají ze standardních operací projekce π, selekce σ, sjednocení $\cup$, průniku $\cap$, rozdílu $-$, spojení $\bowtie$ a nové operace rank μ. Operace rank μ_p nad vstupní relací $R_{\mathcal{P}}$ provádí vyhodnocení ohodnocovací funkce p. Výsledná relace $R_{\mathcal{P} \cup \{p\}}$ obsahuje stejné záznamy jako vstupní relace $R_{\mathcal{P}}$, ale je seřazena dle nového uspořádání, které je dáno ohodnocovacími funkcemi $\mathcal{P} \cup \{p\}$.

Protože relace s ranky mají oproti klasickým relacím navíc vlastnost uspořádání, je třeba u každé operace rank-relační algebry definovat, jakým způsobem aplikace dané operace ovlivňuje toto uspořádání. Unární operace (projekce, selekce) uspořádání neovlivňují. Binární operace mohou probíhat nad relacemi (např. $R_{\mathcal{P}_1}$, $S_{\mathcal{P}_2}$), jejichž uspořádání je dáno obecně různými podmnožinami vyhodnocených ohodnocovacích funkcí ($\mathcal{P}_1, \mathcal{P}_2 \subseteq \{p_1, \ldots, p_n\}$).

Operace	$\vartheta(R_{\mathcal{P}_1}, \dots)$	$t \in \vartheta(R_{\mathcal{P}_1}, \dots) \Leftrightarrow$	$t_1 <_{\vartheta(R_{\mathcal{P}_1}, \dots)} t_2 \Leftrightarrow$		
Rank	$\mu_p(R_{\mathcal{P}})$	$t \in R_{\mathcal{P}}$	$\overline{\mathcal{F}}_{\mathcal{P} \cup \{p\}}[t_1]$	$<$	$\overline{\mathcal{F}}_{\mathcal{P} \cup \{p\}}[t_2]$
Selekce	$\sigma_c(R_{\mathcal{P}})$	$t \in R_{\mathcal{P}}$ a t splňuje c	$\overline{\mathcal{F}}_{\mathcal{P}}[t_1]$	$<$	$\overline{\mathcal{F}}_{\mathcal{P}}[t_2]$
Sjednocení	$R_{\mathcal{P}_1} \cup S_{\mathcal{P}_2}$	$t \in R_{\mathcal{P}_1}$ nebo $t \in S_{\mathcal{P}_2}$	$\overline{\mathcal{F}}_{\mathcal{P}_1 \cup \mathcal{P}_2}[t_1]$	$<$	$\overline{\mathcal{F}}_{\mathcal{P}_1 \cup \mathcal{P}_2}[t_2]$
Průnik	$R_{\mathcal{P}_1} \cap S_{\mathcal{P}_2}$	$t \in R_{\mathcal{P}_1}$ a $t \in S_{\mathcal{P}_2}$	$\overline{\mathcal{F}}_{\mathcal{P}_1 \cup \mathcal{P}_2}[t_1]$	$<$	$\overline{\mathcal{F}}_{\mathcal{P}_1 \cup \mathcal{P}_2}[t_2]$
Rozdíl	$R_{\mathcal{P}_1} - S_{\mathcal{P}_2}$	$t \in R_{\mathcal{P}_1}$ a $t \notin S_{\mathcal{P}_2}$	$\overline{\mathcal{F}}_{\mathcal{P}_1}[t_1]$	$<$	$\overline{\mathcal{F}}_{\mathcal{P}_1}[t_2]$
Spojení	$R_{\mathcal{P}_1} \bowtie_c S_{\mathcal{P}_2}$	$t \in \sigma_c(R_{\mathcal{P}_1} \times S_{\mathcal{P}_2})$	$\overline{\mathcal{F}}_{\mathcal{P}_1 \cup \mathcal{P}_2}[t_1]$	$<$	$\overline{\mathcal{F}}_{\mathcal{P}_1 \cup \mathcal{P}_2}[t_2]$

Tabulka 1.: Tabulka popisuje operace rank-relační algebry. První sloupec obsahuje operace a jejich vstupní relace. Druhý sloupec představuje podmínku, která musí platit, aby byl záznam t zahrnut ve výsledné relaci dané operace. Třetí sloupec popisuje uspořádání výsledné relace. Převzato z [27].

Uspořádání výsledné relace je potom dáno ohodnocovacími funkcemi $\mathcal{P}_1 \cup \mathcal{P}_2$ (s výjimkou operace rozdílu). Definice rozdílu není příliš vhodná – pokud relace menšitele $S_{\mathcal{P}_2}$ obsahuje nějaký záznam s libovolně malým rankem, potom tento záznam nebude zahrnut ve výsledku bez ohledu na rank v relaci menšence $R_{\mathcal{P}_1}$. Tabulka 1. shrnuje operace rank-relační algebry.

Rankování lze díky operaci rank μ považovat za element prvního řádu. Tabulka 2. obsahuje přehled algebraických identit, které se váží k operaci rank. Díky identitě 1 lze rankování ohodnocovacími funkcemi $p_1, \dots, p_n$ rozložit na několik operací rank μ. Dále díky identitám 4 a 5 je možné operaci rank μ přeskupovat s ostatními operacemi. Databázovému systému je tak umožněno provádět optimalizace dotazů s ranky.

Identita 1: Rozdělení rankování $R_{\{p_1,p_2,\dots,p_n\}} \equiv \mu_{p_1}(\mu_{p_2}(\cdots(\mu_{p_n}(R))\cdots))$
Identita 2: Komutativita binárních operací $R_{\mathcal{P}_1} \theta S_{\mathcal{P}_2} \equiv S_{\mathcal{P}_2} \theta R_{\mathcal{P}_1}$, kde $\theta \in \{\cup, \cap, \bowtie_c\}$
Identita 3: Asociativita binárních operací $(R_{\mathcal{P}_1} \theta S_{\mathcal{P}_2}) \theta T_{\mathcal{P}_3} \equiv R_{\mathcal{P}_1} \theta (S_{\mathcal{P}_2} \theta T_{\mathcal{P}_3})$, kde $\theta \in \{\cup, \cap, \bowtie_c\}$
Identita 4: Komutativita operace rank μ $\mu_{p_1}(\mu_{p_2}(R_{\mathcal{P}})) \equiv \mu_{p_2}(\mu_{p_1}(R_{\mathcal{P}}))$ $\sigma_c(\mu_p(R_{\mathcal{P}})) \equiv \mu_p(\sigma_c(R_{\mathcal{P}}))$
Identita 5: Vztah operace rank μ k ostatním operacím $\mu_p(R_{\mathcal{P}_1} \bowtie_c S_{\mathcal{P}_2}) \equiv \mu_p(R_{\mathcal{P}_1}) \bowtie_c S_{\mathcal{P}_2}$, pokud atributy z p jsou pouze v R $\mu_p(R_{\mathcal{P}_1} \bowtie_c S_{\mathcal{P}_2}) \equiv \mu_p(R_{\mathcal{P}_1}) \bowtie_c \mu_p(S_{\mathcal{P}_2})$, pokud atr. z p jsou podmn. atr. v c $\mu_p(R_{\mathcal{P}_1} \cup S_{\mathcal{P}_2}) \equiv \mu_p(R_{\mathcal{P}_1}) \cup S_{\mathcal{P}_2} \equiv \mu_p(R_{\mathcal{P}_1}) \cup \mu_p(S_{\mathcal{P}_2})$ $\mu_p(R_{\mathcal{P}_1} \cap S_{\mathcal{P}_2}) \equiv \mu_p(R_{\mathcal{P}_1}) \cap S_{\mathcal{P}_2} \equiv \mu_p(R_{\mathcal{P}_1}) \cap \mu_p(S_{\mathcal{P}_2})$ $\mu_p(R_{\mathcal{P}_1} - S_{\mathcal{P}_2}) \equiv \mu_p(R_{\mathcal{P}_1}) - S_{\mathcal{P}_2} \equiv \mu_p(R_{\mathcal{P}_1}) - \mu_p(S_{\mathcal{P}_2})$

Tabulka 2.: Přehled algebraických identit, které se váží k rankování [27].

2.1.3. Inkrementální zpracování

Kromě algebraických identit je pro efektivní vyhodnocování dotazů nezbytná podpora pro neblokující inkrementální zpracování. Protože uživatele často zajímá pouze prvních k záznamů, byla by úplná materializace zbytečná a neefektivní. Stejně jako v klasických databázích se využívá tzv. *pipelining*, kdy jednotlivé fyzické operátory nematerializují celou relaci, ale postupně na požádání vracejí další záznamy z výstupní relace daného operátoru.

Protože jsou zpracovávány relace s ranky, musejí fyzické operátory vracet záznamy v pořadí, které je dáno dílčími ranky jednotlivých záznamů. Aby mohl operátor vrátit záznam t_i, musí být zajištěno, že každý další záznam t_j, který tento operátor vrátí v průběhu dalšího zpracování po t_i, nebude mít větší rank než t_i. Musí tedy platit $\overline{\mathcal{F}}_{\mathcal{P}}[t_i] \geq \overline{\mathcal{F}}_{\mathcal{P}}[t_j]$. [27]

Jako příklad autoři uvádějí operátor μ_p realizující výpočet operace rank nad vstupní relací $R_{\mathcal{P}}$, jejíž záznamy jsou postupně načítány z podřízeného operátoru x. Operátor μ_p musí vracet záznamy podle nového uspořádání, které je dáno ohodnocovacími funkcemi $\mathcal{P} \cup \{p\}$. Pro každý záznam t načtený z x operátor μ_p vypočítá nový dílčí rank $\overline{\mathcal{F}}_{\mathcal{P}\cup\{p\}}$ a zařadí tento záznam do prioritní fronty, ve které vyšší rank záznamu znamená jeho vyšší prioritu. Operátor μ_p může vrátit záznam t_{top} ze začátku prioritní fronty, pouze pokud byl z x načten další záznam t', pro který platí $\overline{\mathcal{F}}_{\mathcal{P}\cup\{p\}}[t_{\text{top}}] \geq \overline{\mathcal{F}}_{\mathcal{P}}[t']$. Potom pro každý záznam t'', který bude v budoucnu načten z x platí $\overline{\mathcal{F}}_{\mathcal{P}\cup\{p\}}[t_{\text{top}}] \geq \overline{\mathcal{F}}_{\mathcal{P}}[t'] \geq \overline{\mathcal{F}}_{\mathcal{P}}[t'']$. Navíc z definice dílčího ranku máme $\overline{\mathcal{F}}_{\mathcal{P}}[t''] \geq \overline{\mathcal{F}}_{\mathcal{P}\cup\{p\}}[t'']$. Celkem tedy máme $\overline{\mathcal{F}}_{\mathcal{P}\cup\{p\}}[t_{\text{top}}] \geq \overline{\mathcal{F}}_{\mathcal{P}\cup\{p\}}[t'']$ – dílčí rank žádného záznamu, který operátor μ_p v průběhu dalšího zpracování vrátí, nepřesáhne dílčí rank t. Operátor μ_p tedy může záznam t vrátit.

2.1.4. Implementace

Autoři pro implementaci využili open-source databázový systém PostgreSQL a přidali do něj nativní podporu pro `RankSQL`. Pro tento účel museli autoři upravit interní reprezentaci záznamů, rozšířit parser, plánovací a optimalizační subsystém a efektivně implementovat prioritní frontu a algoritmy pro operátory pracující s ranky [27].

Protože počet záznamů v prioritní frontě může být během výpočtu velký, bylo nutné frontu doplnit o swapování na disk. Z hlediska efektivního zpracování je výhodné odsunout na disk záznamy, jejichž dílčí rank je ve srovnání s ostatními záznamy ve frontě malý. Naopak záznamy s vysokým rankem je vhodné udržovat v paměti, jelikož je pravděpodobné, že je bude brzy možné vypsat na výstup.

Implementace fyzických operátorů je založena na existujících algoritmech [27]. Algoritmus pro operátor realizující rankování μ_p je speciálním případem algoritmů `MPro` [26] a `Upper` [9]. Operátor pro spojení $\bowtie_c$ využívá algoritmy `HRJN` (hash rank-join) a `NRJN` (nested-loop rank-join) [30] [31].

2.2. Přístupy založené na fuzzy logice v širokém slova smyslu – SQLf

Základem těchto přístupů jsou fuzzy relace v Zadehově pojetí fuzzy logiky, kde strukturu pravdivostních hodnot tvoří interval $[0, 1] \subseteq \mathbb{R}$. Stupeň příslušnosti záznamu do relace vyjadřuje, do jaké míry daný záznam vyhovuje tzv. fuzzy predikátům v dotazu. Fuzzy predikáty slouží k formalizaci flexibilních dotazů a v dotazech nahrazují klasické podmínky pro selekci nebo spojení [36].

Fuzzy predikáty je možné vyjádřit pomocí fuzzy množin a fuzzy relací. Z jednotlivých fuzzy predikátů je možné sestavovat složené podmínky pomocí spojek konjunkce, disjunkce a negace. Tyto spojky jsou často interpretovány pomocí t-norem, co-norem a involutivní negace $\neg a = 1 - a$.

Autoři se v [36] věnují dvěma dotazovacím jazykům, které umožňují pokládat flexibilní dotazy. Nejprve popisují rozšířenou relační algebru, poté se věnují jazyku `SQLf`, který je určen uživatelům databázového systému s flexibilními dotazy.

2.2.1. Rozšířená relační algebra – přehled

V rozšířené relační algebře jsou klasické relace nahrazeny fuzzy relacemi. Rozšířená relační algebra neobsahuje žádné nové operace. Stávající operace jsou přizpůsobeny pro práci s fuzzy relacemi následovně [36]. Kartézský součin fuzzy relací r a s nad relačními schématy R a S je definován jako:

$$r \times s = \{{}^{\mu}/_{uv} \mid u \in \text{support}(r), v \in \text{support}(s), \mu = \min(\mu_r(u), \mu_s(v))\},$$

Pro fuzzy relace r a s nad stejným relačním schématem jsou definovány množinové operace sjednocení, průniku a rozdílu takto:

$$\begin{aligned}
r \cup s = &\{{}^{\mu}/_{u} \mid u \in \text{support}(r), u \notin \text{support}(s), \mu = \mu_r(u)\} \cup \\
&\{{}^{\mu}/_{u} \mid u \in \text{support}(s), u \notin \text{support}(r), \mu = \mu_s(u)\} \cup \\
&\{{}^{\mu}/_{u} \mid u \in \text{support}(r), u \in \text{support}(s), \mu = \max(\mu_r(u), \mu_s(v))\} \\
r \cap s = &\{{}^{\mu}/_{u} \mid u \in \text{support}(r), u \in \text{support}(s), \mu = \min(\mu_r(u), \mu_s(v))\} \\
r - s = &\{{}^{\mu}/_{u} \mid u \in \text{support}(r), \mu = \min(\mu_r(u), 1 - \mu_s(u))\}
\end{aligned}$$

Operace selekce $\sigma_\psi(r)$, kde ψ je fuzzy predikát; spojení $r \bowtie_\theta s$, kde r je nad schématem RS, s nad ST a θ je fuzzy komparátor (binární fuzzy relace nad $S \times S$); a projekce $\pi_S(r)$, kde r je nad R a $S \subseteq R$ jsou definovány takto:

$$\begin{aligned}
\sigma_\psi(r) &= \{{}^{\mu}/_{u} \mid u \in \text{support}(r), \mu = \min(\mu_r(u), \mu_\psi(u))\} \\
r \bowtie_\theta s &= \{{}^{\mu}/_{uv} \mid u \in \text{support}(r), v \in \text{support}(s), \\
&\qquad \mu = \min(\mu_r(u), \mu_s(v), \mu_\theta(u[S], v[S])\} \\
\pi_S(r) &= \{{}^{\mu}/_{v} \mid \exists u \in \text{support}(r) : u[S] = v, \\
&\qquad \mu = \sup\{\mu_r(u) \mid u \in \text{support}(r), u[S] = v\}\}
\end{aligned}$$

Jelikož lze klasické (crisp) relace, které slouží k uchovávání dat v databázovém systému, považovat za speciální případ fuzzy relací, je možné operace rozšířené relační algebry s těmito relacemi bez problému používat.

2.2.2. Rozšířená relační algebra – dělení

Speciální kapitolu věnují autoři operaci relačního dělení [36]. Vycházejí z původní Coddovy definice dělení [13], která je založena na pojmu *image set*. Nechť je $\mathcal{D}$ relace nad relačním schématem XY a $x \in \mathrm{Tupl}(X)$. Pod pojmem *image set* rozumíme množinu

$$\Gamma_{\mathcal{D}}^{-1}(x) = \{y \in \mathrm{Tupl}(Y) \mid xy \in \mathcal{D}\}.$$

Potom pro relaci $\mathcal{D}_1$ (dělenec) nad schématem RS a relaci $\mathcal{D}_2$ (dělitel) nad schématem ST je klasická (crisp) operace relačního dělení definována takto [36]:

$$\mathcal{D}_1 \div \mathcal{D}_2 = \{r \in \mathrm{Tupl}(R) \mid \pi_S(\mathcal{D}_2) \subseteq \Gamma_{\mathcal{D}_1}^{-1}(r)\}$$

Vyjádřením vztahu „býti podmnožinou“ pomocí logické formule dostaneme alternativní definici [36]:

$$\mathcal{D}_1 \div \mathcal{D}_2 = \{r \in \mathrm{Tupl}(R) \mid \forall s \in \mathrm{Tupl}(S)\colon s \in \pi_S(\mathcal{D}_2) \Rightarrow rs \in \mathcal{D}_1\}$$

Obě definice nejsou z pohledu relačních databází zcela korektní, protože připouštějí jako výsledek nekonečnou relaci, a to v případě prázdné relace dělitele. Zobecněné definice dělení pro rozšířenou relační algebru tímto problémem netrpí, jelikož jsou definovány nad $\pi_R(\mathcal{D}_1)$ namísto $\mathrm{Tupl}(R)$. Je pravděpodobné, že autoři tento fakt implicitně předpokládají i u obou předchozích definic. Tento přístup však s sebou nese jiný problém, kdy při prázdné relaci dělitele výsledek neodpovídá zamýšlenému logickému významu operace dělení. Tímto problémem se práce podrobněji zabývá v kapitole 4. (TODO: včetně strany!).

Autoři zavádějí operaci dělení v rozšířené relační algebře přímým zobecněním výše uvedené definice, kdy uvažují relaci „býti podmnožinou“ jako fuzzy relaci. Autoři dále uvádějí, že je možné ekvivalentně operaci dělení zavést zobecněním druhé (alternativní) definice takto

$$\mathcal{D}_1 \div \mathcal{D}_2 = \{{}^{\mu}\!/_{r} \mid r \in \pi_R(\mathrm{support}(\mathcal{D}_1)), \mu = \min_{s \in \mathrm{Tupl}(S)} \left(\mu_{\pi_S(\mathcal{D}_2)}(s) \Rightarrow \mu_{\mathcal{D}_1}(rs)\right)\}$$

a to za předpokladu, že pokud je ve výše uvedeném vztahu použita

- R-implikace (*residuovaná implikace*), pak musí být kartézský součin[1] založen na t-normě, se kterou je daná implikace provázána prostřednictvím podmínky adjunkce. Tato podmínka v podstatě odpovídá použití standardní fuzzy logiky v úzkém slova smyslu.

[1] Autoři vyjma výše uvedené definice kartézského součinu založeného na operaci minima připouštějí i obecnější variantu, ve které je možné nahradit minimum jinou konjunktivní operací.

- S-implikace (*strong implikace*), pak musí být kartézský součin založen na nekomutativní konjunkci příslušné dané S-implikaci.

V neposlední řadě autoři uvádějí, za jakých podmínek je možné dělení považovat za odvozenou operaci. Dělení je možné vyjádřit pomocí známé definice založené na kartézském součinu a dvojí aplikaci rozdílu, pokud jsou splněny všechny následující předpoklady: [36]:

- kartézský součin je definován pomocí minimové t-normy,
- první dva výskyty relace dělence jsou nahrazeny za *support* této relace,
- operace dělení je založena na S-implikaci, nebo pokud je založena na R-implikaci, pak je nutné uvažovat operaci rozdílu založenou na nekomutativní konjunkci.

Dělení lze potom vyjádřit jako:

$$\mathcal{D}_1 \div \mathcal{D}_2 = \pi_R(\text{support}(\mathcal{D}_1)) - \pi_R\left(\text{support}(\mathcal{D}_1) - (\pi_R(\mathcal{D}_1) \times \pi_S(\mathcal{D}_2))\right)$$

Použití nekomutativní konjunkce není nijak zdůvodněno. Není jasné, jakým způsobem interpretovat operace relační algebry založené na nekomutativních spojkách. Navíc vyvstává otázka, zdali je správné takové operace nazývat stejným jménem jako jejich protějšky definované pomocí komutativních spojek, jelikož mohou mít odlišné a neočekávané vlastnosti.

2.2.3. S-implikace

Autoři nejen ve výše uvedených definicích uvažují tzv. S-implikace. Jedná se o alternativní zavedení implikace, jež je založeno na zákonu klasické výrokové logiky, který umožňuje definovat implikaci prostřednictvím konjunkce a negace jako $\varphi \to \psi = \neg(\varphi \wedge \neg\psi)$.

Oproti R-implikacím se však s S-implikacemi pojí problém s korektností pravidla odloučení [25], které tvoří základní kámen klasické logiky a jejích zobecnění. Nutnou podmínkou pro korektnost logiky postavené na S-implikacích je fakt, že v dané logice musí platit zákon dvojí negace ($\varphi = \neg\neg\varphi$). Toto tvrzení nyní prokážeme.

Nechť je $\mathbf{L} = \langle L, \otimes, 1, \neg\rangle$ algebra, kde $\otimes$ je binární asociativní operace, $1 \in L$ je největší prvek L a současně neutrální prvek binární operace $\otimes$. Na unární operaci $\neg$ neklademe žádné podmínky. Dále uvažujeme odvozenou binární operaci $a \to b = \neg(a \otimes \neg b)$. Pomocí operace $\otimes$ interpretujeme logickou konjunkci, pomocí $\neg$ negaci a pomocí $\to$ S-implikaci.

Odvozovacím pravidlem klasické výrokové logiky je pravidlo odloučení (*modus ponens*), které formalizuje elementární krok usuzování a je ve známém tvaru

$$\frac{\varphi \Rightarrow \psi, \varphi}{\psi}.$$

Výroková logika je korektní, nutně tedy musí být korektní i pravidlo odloučení. Lze uvažovat „sémantickou“ verzi tohoto pravidla

$$\frac{\|\varphi \Rightarrow \psi\| = 1, \|\varphi\| = 1}{\|\psi\| = 1},$$

kde notací $\|\cdot\|$ rozumíme pravdivostní hodnotu dané formule (v modelu). V zobecnění výrokové logiky, které využívá více stupňů pravdivostních hodnot, můžeme tuto sémantickou verzi pravidla odloučení uvažovat ve tvaru

$$\frac{\|\varphi \Rightarrow \psi\| \geq a, \|\varphi\| \geq b}{\|\psi\| \geq a \otimes b}.$$

Význam $\|\cdot\|$ zůstává nezměněn. Hodnoty $a, b \in L$ představují dolní mez pravdivostní hodnoty příslušné formule v daném modelu. Pravidlo nám říká, že je-li $\varphi \Rightarrow \psi$ pravdivé alespoň ve stupni a a φ ve stupni b, pak můžeme odvodit, že formule ψ je platná alespoň ve stupni $a \otimes b$, ale ne více než je tato hodnota. Nemůžeme odvodit pravdivostní hodnotu důsledku vyšší než mají samy předpoklady – tuto podmínku lze nazvat jako korektnost odvozovacího pravidla. Máme tedy

$$\text{pokud } a \leq \|\varphi \Rightarrow \psi\| \text{ a } b \leq \|\varphi\|, \text{ potom } a \otimes b \leq \|\psi\|.$$

Výroková logika i její zobecnění ctí tzv. princip kompozicionality, kdy pravdivostní hodnota složeného výroku je dána aplikací příslušné operace interpretující danou spojku na pravdivostní hodnoty formulí, ze kterých se daný výrok skládá. Nechť $b = \|\varphi\|$ a $c = \|\psi\|$. Potom z principu kompozicionality dostaneme $\|\varphi \Rightarrow \psi\| = \|\varphi\| \to \|\psi\| = b \to c = \neg(b \otimes \neg c)$. Výše uvedenou podmínku lze nyní zapsat takto:

$$\text{Pokud } a \leq \neg(b \otimes \neg c), \text{ potom } a \otimes b \leq c.$$

Tato podmínka musí platit i pro $a = \neg(b \otimes \neg c)$ a $b = 1$:

$$\begin{aligned} a \otimes b &\leq c \\ \neg(b \otimes \neg c) \otimes b &\leq c \\ \neg(1 \otimes \neg c) \otimes 1 &\leq c \\ \neg\neg c &\leq c \end{aligned}$$

Z druhé strany lze uvažovat podmínku maximální síly odvozovacího pravidla. Pokud odvodíme, že formule ψ je platná alespoň ve stupni $a \otimes b$, potom požadujeme, aby $a \leq \|\varphi \Rightarrow \psi\|$ a $b \leq \|\varphi\|$. Tedy dolní mez $a \otimes b$ je maximální možná. Pro $b = \|\varphi\|$ a $c = \|\psi\|$ lze podmínku zapsat jako:

$$\text{Pokud } a \otimes b \leq c \text{ potom } a \leq \neg(b \otimes \neg c).$$

Podmínka musí platit i pro $a \otimes b = c$ a $b = 1$:

$$\begin{aligned} a &\leq \neg(b \otimes \neg c) \\ a &\leq \neg(b \otimes \neg(a \otimes b)) \\ a &\leq \neg(1 \otimes \neg(a \otimes 1)) \\ a &\leq \neg\neg a \end{aligned}$$

Aby měla logika založená na S-implikacích korektní a maximálně silné pravidlo odloučení, je nutné, aby operace negace splňovala zákon dvojí negace ($a = \neg\neg a$). Při použití S-implikací se tak velmi zužuje třída fuzzy logik, které je možné uvažovat. Existuje celá řada fuzzy logik, které nesplňují zákon dvojí negace (Gödelova, produktová, ...).

2.2.4. Jazyk SQLf – přehled

Cílem autorů bylo vytvořit dotazovací jazyk, který by umožnil pokládat flexibilní dotazy a přitom byl co nejpodobnější jazyku `SQL` [36]. Základním stavebním kamenem jazyka `SQLf` je stejně jako v `SQL` dotaz typu projekce-selekce-spojení. Protože výsledkem dotazu je fuzzy množina a uživatele často zajímají pouze záznamy, které vyhovují dotazu nejlépe (jejich stupeň příslušnosti do výsledné relace je nejvyšší), umožňuje jazyk `SQLf` uživateli navíc specifikovat kolik nejlepších záznamů si přeje zobrazit, případně jaký minimální stupeň příslušnosti musejí zobrazené záznamy mít. Tento dotaz se v jazyce `SQLf` zapíše takto [36]

$$\textbf{select}\ [n \mid t \mid n, t]\ \langle\text{atributy}\rangle\ \textbf{from}\ \langle\text{relace}\rangle\ \textbf{where}\ \langle\text{podmínky}\rangle,$$

kde uživatel může volitelně specifikovat maximální počet záznamů n, minimální stupeň příslušnosti t, případně obě hodnoty. Dále uživatel zadá požadované atributy, vstupní relace a jednu či více fuzzy nebo klasických podmínek.

Fuzzy podmínky mohou být obecně dvojího typu. Prvním typem je podmínka, kdy zjišťujeme v jakém stupni leží hodnota atributu daného záznamu ve fuzzy množině, která formalizuje tuto podmínku. Příkladem může být podmínka *age* **is** *young*. Druhý typ podmínky je založen na fuzzy relacích, kde pro dvojici hodnot zjišťujeme jejich stupeň příslušnosti v dané relaci, například *salary* ≈ 3000.

Jazyk `SQLf` podporuje vnořené dotazy, které je možné vytvářet pomocí klíčových slov **in**, **exists**, aj. V případě konstrukce vnořeného dotazu pomocí **in** je sémantika následující [36]. Protože autoři připouštějí (stejně jako `SQL`) duplicitní záznamy v relacích, musejí definovat stupeň příslušnosti (vnějšího) záznamu t do fuzzy relace $\mathcal{D}$ vzniknuvší vyhodnocením vnořeného dotazu jako

$$\mu_{(t\ \textbf{in}\ \mathcal{D})} = \bigvee_{t' \in \text{support}(\mathcal{D})} \min(\mu_{=}(t, t'), \mu_{\mathcal{D}}(t'))$$

Jazyk `SQLf` dále podporuje množinové operace, jejichž syntaxe je totožná s jazykem `SQL` a sémantika vychází z rozšířené relační algebry. V operacích, jako je množinový rozdíl či **not exists**, se vždy používá involutivní negace $\neg a = 1 - a$. V neposlední řadě jazyk `SQLf` podporuje různé typy fuzzy kvantifikátorů, které slouží k modelování výrazů jako *pro většinu platí*, *alespoň pro polovinu platí*, atd.

2.2.5. Jazyk SQLf – podpora pro definici fuzzy podmínek

Dalším důležitým cílem jazyka `SQLf` je uživateli poskytnout nástroje pro snadné zadávání fuzzy podmínek – fuzzy množin. Podle autorů se v praxi často používají lichoběžníkové fuzzy množiny, které je možné charakterizovat pomocí čtyř hodnot $F = (A, B, a, b)$, kde interval $[A, B]$ udává tzv. jádro fuzzy množiny (hodnoty plně náležící do této fuzzy množiny) a interval $[A-a, B+b]$ představuje support fuzzy množiny (hodnoty náležící do fuzzy množiny v nenulovém stupni).

Jazyk `SQLf` disponuje několika způsoby pro zadávání fuzzy podmínek [36]. Základní možností jsou příkazy **create fuzzy predicate** a **create comparator**. Alternativní způsob nabízí uživatelům zakomponovat fuzzy podmínky přímo do dotazu. Za předpokladu využití lichoběžníkové fuzzy množiny stačí pro numerické atributy definovat *ideální* hodnoty (odpovídají jádru fuzzy množiny) a *přijatelné* hodnoty (odpovídají supportu množiny). Jako příklad použití alternativního způsobu definice fuzzy podmínek autoři uvádějí tento dotaz (zde uveden ve zjednodušené verzi):

select $*$ **from** cars **where** (*color* **is** C_1) **and** (*price* **is** *low_P*)
with preferences
C_1: $\{{}^{1}/_{\text{blue}}, {}^{1}/_{\text{black}}, {}^{0.8}/_{\text{green}}, {}^{0.7}/_{\text{gray}}, {}^{0.5}/_{\text{red}}, {}^{0.3}/_{\text{orange}}\}$,
low_P: $\langle$**ideal:** $price \leq 4\,000$, **acceptable:** $price \leq 6\,000\rangle$

2.2.6. Jazyk SQLf – implementace

Implementace jazyka `SQLf` se uvažuje ve formě dodatečné vrstvy nad existujícím databázovým systémem a v [36] jsou popsány teoretické aspekty implementace (algoritmy, aj.). Podle typu dotazu se pro zpracování používají buď specializované algoritmy, anebo se využívá tzv. *princip odvození* [36].

Tento princip předpokládá, že je pro daný dotaz specifikován minimální stupeň příslušnosti záznamů do výsledné relace (ozn. α). Na základě této hodnoty α se dotaz v jazyce `SQLf` převede na standardní `SQL` dotaz a předloží se databázovému systému. Výsledkem je klasická relace, která tvoří nadmnožinu α-řezu fuzzy relace, jež představuje odpověď na původní fuzzy dotaz. Tato klasická relace je dále zpracována a je z ní vypočtena požadovaná odpověď. Efektivita *principu odvození* záleží na typu dotazu, počtu podmínek a typu spojek.

Experimentální implementace je postavena nad systémem `Oracle 8i` a poskytuje API pro programovací jazyky jako jsou ASP, JSP, Java [23].

3. Přístup založený na fuzzy logice v úzkém slova smyslu

Coddův relační model je založen na klasické dvouhodnotové logice. Stejně tak jsou na této logice založeny i základní dotazovací jazyky – jazyk založený na relační algebře a relační kalkul. Hlavní podíl na úspěchu Coddova relačního modelu se připisuje jeho koncepční čistotě, která vychází právě z úzkého vztahu k logice. Díky tomuto vztahu jej navíc bylo možné do hloubky prozkoumat a umožnit tak jeho efektivní implementaci. [3] [21]

Hlavní myšlenkou přístupu založeného na fuzzy logice v úzkém slova smyslu [3] je uvažovat zobecnění Coddova relačního modelu, které získáme tak, že místo klasické dvouhodnotové logiky model založíme na vícehodnotové fuzzy logice. Takto, při zachování koncepční čistoty, ve zobecněném modelu přirozeně vyplynou pojmy, jako je podobnost. Můžeme hovořit o tom, že nějaká n-tice hodnot splňuje dotaz v určitém stupni (na rozdíl od klasického modelu, kde n-tice buď dotazu vyhovuje, nebo ne). [3]

Právě díky zachování vztahu k logice je možné zobecněný model podrobit důkladnému zkoumání stejně jako klasický relační model. Tento fakt představuje velkou výhodu oproti různým rozšířením relačního modelu, které často tento vztah postrádají. Navíc takto získaný model je skutečným zobecněním klasického Coddova relačního modelu – klasický model je speciálním případem zobecněného.

Jelikož tato práce vychází z přístupu [3] a dále jej rozšiřuje, je tato kapitola věnována podrobnějšímu představení zobecněného modelu včetně obou základních dotazovacích jazyků.

3.1. Struktura pravdivostních hodnot

Oproti klasické logice, ve které je každá formule buď zcela pravdivá, nebo zcela nepravdivá, nám fuzzy logika umožňuje uvažovat i mezilehlé stupně pravdivosti. Říkáme, že formule je pravdivá v nějakém stupni. Klasická logika je založena na dvouprvkové Booleově algebře. Pro fuzzy logiku potřebujeme obecnější strukturu pravdivostních hodnot – vhodnou strukturou [2] je úplný reziduovaný svaz $\mathbf{L} = \langle L, \wedge, \vee, \otimes, \rightarrow, 0, 1 \rangle$, kde

- $\langle L, \wedge, \vee, 0, 1 \rangle$ je úplný svaz, jehož nejmenší (největší) prvek je 0 (1),
- $\langle L, \otimes, 1 \rangle$ tvoří komutativní monoid,
- operace součinu $\otimes$ a residua $\rightarrow$ splňují podmínku adjunkce

$$a \otimes b \leq c \Leftrightarrow a \leq b \rightarrow c$$

Množina L obsahuje uvažované pravdivostní hodnoty. Pomocí operací infima $\wedge$ a suprema $\vee$ interpretujeme všeobecný a existenční kvantifikátor. Operace součinu $\otimes$ a residua $\rightarrow$ slouží k interpretaci logických spojek konjunkce a implikace.

Důležitým pojmem je zobecnění klasické množiny – (fuzzy) **L**-množina. Množinu lze chápat jako zobrazení, které každému prvku z uvažovaného univerza X přiřazuje pravdivostní hodnotu, ve kterém tento prvek do dané množiny náleží [2]. Klasická množina pracuje pouze se dvěma pravdivostními hodnotami – prvek buď do množiny patří, nebo ne. **L**-množina umožňuje jemnější pohled. Formálně, nechť je $\mathbf{L}$ úplný residuovaný svaz a X neprázdná množina (universum), potom **L**-množina A v X je zobrazení $A\colon X \to L$. Pro $x \in X$ hodnotu $A(x)$ nazýváme stupeň příslušnosti x do A. [2]

3.2. Základní definice

Ústředním pojmem relačního modelu je relace, která formalizuje intuitivní pojem tabulky s daty. S relacemi je potom možné manipulovat pomocí operací relační algebry. Aby bylo možné zavést pojem relace, je nejdříve nutné definovat několik základních pojmů.

3.2.1. Atributy, typy, domény

Uvažujeme syntaktický pojem *atribut*, který slouží jako symbolické jméno pro sloupce tabulky. Množinu všech atributů označujeme Y a předpokládáme, že je tato množina spočetná. Libovolnou konečnou podmnožinu $R \subseteq Y$ nazýváme *relační schéma*. Dále uvažujeme spočetnou množinu $\mathfrak{C}$ objektových konstant, které reprezentují hodnoty, jež se mohou objevit v tabulkách. [3]

Dále uvažujeme syntaktický pojem *typ*, který slouží jako pojmenování množiny přípustných hodnot. Všechny atributy a objektové konstanty mají přiřazeny svůj typ, čímž je zajištěna typová kompatibilita – není možné definovat dvě relace, jejichž relační schéma by obsahovalo atribut stejného jména, ale jiného typu. Jinými slovy, atributy či objektové konstanty stejného jména jsou vždy stejného typu. Přiřazení typů je formalizováno pomocí *typové deklarace*. Typovou deklarací pro Y a $\mathfrak{C}$ rozumíme strukturu $\mathbf{\Lambda} = \langle \Lambda, \lambda \rangle$, kde Λ je neprázdná množina typů a $\lambda\colon Y \cup \mathfrak{C} \to \Lambda$ je zobrazení takové, že pro každý typ $\tau \in \Lambda$ existuje nekonečná podmnožina atributů $Y' \subseteq Y$, pro kterou platí $\lambda(Y') = \{\tau\}$. [3]

Sémantickým protějškem pojmu typ je *doména*. Doménou D_y rozumíme neprázdnou množinu všech přípustných hodnot, kterých může daný atribut $y \in Y$ nabývat. V klasickém případě se každá doména uvažuje s relací rovnosti, aby bylo možné prvky z domény vzájemně porovnávat na rovnost, což je možné považovat za krajní případ podobnosti. Prvky si jsou buď zcela podobné (jsou stejné), nebo si nejsou vůbec podobné. V zobecněném modelu jsou domény k tomuto účelu vybaveny relacemi podobnosti, díky kterým lze uvažovat i mezilehlé stupně podobnosti. Z logického pohledu pomocí relací podobnosti interpretujeme příslušné relační symboly ve formulích relačního kalkulu a relační algebry.

Relací podobnosti rozumíme každou **L**-relaci $\approx_y \colon D_y \times D_y \to L$, která je [3]:

- reflexivní – pro každý prvek $a \in D_y$ máme $a \approx_y a = 1$,
- symetrická – pro všechny $a, b \in D_y$ máme $a \approx_y b = b \approx_y a$.

Ačkoliv existují nesymetrické podobnostní míry (Tversky index, který kvantifikuje podobnost objektu k prototypu, a další [1][41][42]), či podobnost v přirozeném jazyce vykazuje v určitých kontextech nesymetričnost (přirovnání, metaforická podobnost [35]), zobecněný model vyžaduje, aby relace podobnosti byla symetrická. Absence této vlastnosti by vedla k nepříznivým důsledkům z pohledu relačního modelu, např. podobnostní spojení by nebylo komutativní [3].

Často je vhodné, aby relace podobnosti splňovala další vlastnosti [3]:

- separabilita – pro každé $a, b \in D_y$ platí $a \approx_y b = 1$ právě když a a b jsou identické (tedy v dané doméně nerozlišitelné)
- $\otimes$-tranzitivita – pro všechny $x, y, z \in D_y$ máme $x \approx y \otimes y \approx z \leq x \approx z$.

Doménu D_y vybavenou relací podobnosti $\approx_y$ označujeme $\langle D_y, \approx_y \rangle$ a nazýváme ji *doména s podobností*. Aby byla zajištěna konzistence mezi typy a doménami, požadujeme, aby domény s podobnostmi $\{\langle D_y, \approx_y \rangle \mid y \in Y\}$ *respektovaly* typovou deklaraci $\mathbf{\Lambda} = \langle \Lambda, \lambda \rangle$ – pro každé $y_1, y_2 \in Y$ musí platit následující. Pokud jsou y_1 a y_2 stejného typu $\lambda(y_1) = \lambda(y_2)$, potom musejí příslušné domény s podobnostmi $\langle D_{y_1}, \approx_{y_1} \rangle$ a $\langle D_{y_2}, \approx_{y_2} \rangle$ splývat. [3]

3.2.2. Obecný kartézský součin, n-tice, relace

Stejně jako v klasickém modelu je pojem relace založen na *obecném kartézském součinu*. Pro indexový systém množin $\{A_i \mid i \in I\}$ definujeme obecný kartézský součin $\prod_{i \in I} A_i$ jako množinu všech zobrazení $f \colon I \to \bigcup_{i \in I} A_i$ takových, že pro každé $i \in I$ máme $f(i) \in A_i$. Pro relační schéma $R \subseteq Y$ uvažujeme obecný kartézský součin $\prod_{y \in R} D_y$, jenž slouží jako universum pro relace nad tímto relačním schématem a je zvykem jej označovat jako $\mathrm{Tupl}(R)$. Prvky této množiny $r \in \mathrm{Tupl}(R)$ nazýváme n-tice nad R, jednotlivé složky n-tic $r(y) \in D_y$ potom nazýváme hodnotami atributu y v n-tici r. [3]

Nyní již můžeme přistoupit k definici pojmu relace v zobecněném modelu. Uvažujme relační schéma $R \subseteq Y$ a dále mějme domény s podobnostmi $\langle D_y, \approx_y \rangle$ příslušné atributům $y \in R$. Potom relací s ranky na R nad doménami s podobnostmi $\{\langle D_y, \approx_y \rangle \mid y \in R\}$ rozumíme každé zobrazení

$$\mathcal{D} \colon \prod_{y \in R} D_y \to L$$

takové, že množina n-tic $\{r \in \prod_{y \in R} D_y \mid \mathcal{D}(r) > 0\}$ je konečná. Stupeň příslušnosti n-tice $r \in \mathrm{Tupl}(R)$ do relace $\mathcal{D}(r)$ nazveme *rank* r v $\mathcal{D}$. [3]

V případě, kdy pro všechny $r \in \mathrm{Tupl}(R)$ máme $\mathcal{D}(r) \in \{0, 1\}$, anebo jako strukturu pravdivostních hodnot $\mathbf{L}$ uvažujeme dvouhodnotovou Booleovu algebru, jsou relace s ranky ve vzájemně jednoznačném vztahu s relacemi z klasického modelu. [3] Pro relaci s ranky $\mathcal{D}$ a klasickou relaci $\mathcal{D}_\mathcal{C}$ v tomto případě platí, že rank r v $\mathcal{D}$ je roven hodnotě charakteristické funkce relace $\mathcal{D}_\mathcal{C}$ pro r.

Důležitým aspektem je interpretace ranků. Obecně v relačních modelech slouží relace kromě uchovávání dat (bázové relace) k reprezentaci výsledků dotazu. V klasickém modelu pro uvažovanou n-tici platí, že buď dotazu plně vyhovuje a je zařazena do výsledné relace – charakteristická funkce relace pro tuto n-tici nabývá hodnoty 1, v zobecněném modelu by měla rank 1 – anebo dotazu nevyhovuje a do výsledné relace zařazena není – charakteristická funkce nabývá hodnotu 0, stejně tak i rank. V zobecněném modelu může být podobnostní dotaz splněn v různých stupních, proto rank $\mathcal{D}(r)$ interpretujeme jako stupeň, ve kterém daná n-tice vyhovuje dotazu [3].

Zvláštní roli hrají relace nad prázdným relačním schématem. Universum $\mathrm{Tupl}(\emptyset)$ obsahuje jediný prvek – prázdnou n-tici. Relace nad prázdným relačním schématem potom odpovídají konstantním zobrazením do množiny pravdivostních hodnot – máme vzájemně jednoznačný vztah mezi relacemi nad prázdným relačním schématem a pravdivostními hodnotami logiky, nad kterou je daný model postaven. V klasickém modelu založeném na dvouhodnotové logice existují dvě relace nad prázdným relačním schématem a jsou jimi prázdná relace `TABLE_DUM` a relace obsahující prázdnou n-tici `TABLE_DEE` [17]. Relace `TABLE_DUM` odpovídá logické nepravdě a chová se jako anihilátor vzhledem k operaci spojení, `TABLE_DEE` potom odpovídá logické pravdě a chová se jako neutrální prvek vzhledem k operaci spojení [17]. V zobecněném modelu existuje relací nad prázdným relačním schématem obecně více. Označujeme je $a_\emptyset$ pro $a \in L$ a definujeme je jako relace, pro které platí $a_\emptyset(\emptyset) = a$. Relacím `TABLE_DUM` a `TABLE_DEE` odpovídají relace $0_\emptyset$, resp. $1_\emptyset$. [3]

3.2.3. Instance databáze jako struktura pro jazyk

Obecně lze dotazy podle [3] chápat jako zobrazení, která transformují vstupní relace na výsledné relace. Dotazování probíhá nad databází – objektem, který zastřešuje několik (bázových) relací, které slouží k perzistentnímu uchování dat a jejich podoba se může v průběhu času vlivem modifikací dat měnit. Vstupní relace dotazů v okamžiku jeho vyhodnocování tedy reprezentují aktuální stav databáze, nad kterou je dotaz pokládán [3].

Pro formalizaci aktuálního stavu databáze uvažujeme následující pojmy. Nejprve zavádíme syntaktický pojem *databázové schéma*, jenž obsahuje názvy a relační schémata relací, které uvažovaná databáze zastřešuje. Formálně, databázové schéma nad Y chápeme jako dvojici $\langle \mathbb{R}, \varrho \rangle$, kde $\mathbb{R}$ je konečná neprázdná množina relačních symbolů a zobrazení $\varrho\colon \mathbb{R} \to 2^Y$ přiřazuje každému relačnímu symbolu $\mathbb{r} \in \mathbb{R}$ relační schéma $\varrho(\mathbb{r}) \subseteq Y$. [3]

Dále uvažujeme tzv. *instanci databáze*, jež reprezentuje aktuální stav uvažované databáze. Instance databáze $\boldsymbol{\mathcal{D}}$ nad databázovým schématem $\langle \mathbb{R}, \varrho \rangle$ nad Y a objektovými konstantami $\mathfrak{C}$ rozumíme trojici $\boldsymbol{\mathcal{D}} = \langle U^{\boldsymbol{\mathcal{D}}}, \mathbb{R}^{\boldsymbol{\mathcal{D}}}, \mathfrak{C}^{\boldsymbol{\mathcal{D}}} \rangle$, kde

- $U^{\boldsymbol{\mathcal{D}}} = \{\langle D_y, \approx_y \rangle \mid y \in Y\}$ je množina domén s podobnostmi,
- $\mathbb{R}^{\boldsymbol{\mathcal{D}}}$ je množina relací, ve které pro každý relační symbol $\mathbb{r} \in \mathbb{R}$ máme relaci $\mathbb{r}^{\boldsymbol{\mathcal{D}}} \in \mathbb{R}^{\boldsymbol{\mathcal{D}}}$ nad relačním schématem $\varrho(\mathbb{r})$ a příslušnými doménami $\{\langle D_y, \approx_y \rangle \mid y \in \varrho(\mathbb{r})\} \subseteq U^{\boldsymbol{\mathcal{D}}}$,
- $\mathfrak{C}^{\boldsymbol{\mathcal{D}}} = \{\mathfrak{c}^{\boldsymbol{\mathcal{D}}} \in \bigcup_{y \in Y} D_y \mid \mathfrak{c} \in \mathfrak{C}\}$ je množina hodnot interpretujících objektové konstanty.

Z logického pohledu je instance databáze strukturou pro jazyk, jenž je dán databázovým schématem a množinou objektových konstant. Základní součást dotazů nad uvažovaným databázovým schématem tvoří relační symboly z tohoto schématu a právě pomocí instance databáze tyto relační symboly interpretujeme. [3]

3.3. Relační algebra

Relační algebra sestává z operací, jimiž je možné manipulovat s relacemi. Na relační algebře je potom založen dotazovací jazyk, ve kterém dotazy tvoříme skládáním relačních operací. Výsledek dotazu pak získáme jeho vyhodnocením v instanci databáze, pomocí které interpretujeme relační symboly a objektové konstanty, které se v dotazu vyskytují [3].

Základ relační algebry tvoří pečlivě vybrané operace, jejichž společná vyjadřovací síla stačí k pokrytí všech dotazů, jež je možné vyjádřit v relačním kalkulu s využitím plné síly predikátové logiky. Prokazuje se tzv. relační úplnost, tedy, že relační algebra i relační kalkul jsou z hlediska vyjadřovací síly rovnocenné. Vzájemnému vztahu relační algebry a relačního kalkulu se věnuje podkapitola 3.5. (str. 37).

Relační algebra pro zobecněný model je koncepčně podobná klasickému přístupu, nicméně musí brát v potaz charakter obecnější logiky, ve které přestávají platit mnohé zákony klasické dvouhodnotové logiky – kupříkladu obecně neplatí zákon dvojí negace. Důsledkem je například nutnost zahrnout operaci relačního dělení do základních operací relační algebry. [3] Operace relačního dělení v jistém smyslu představuje všeobecnou kvantifikaci. Bez zákona dvojí negace však nejsme schopni vyjádřit všeobecný kvantifikátor pomocí existenčního kvantifikátoru známým vztahem $(\forall x)\varphi \equiv \neg(\exists x)\neg\varphi$, a tak zavést operaci dělení pomocí jiných operací relační algebry.

Všechny operace relační algebry jsou v [3] vždy definovány tak, aby s podobnostmi a ranky pracovaly logicky korektním způsobem, jejich výsledkem byla konečná relace a aby tvořily konzervativní rozšíření klasické relační algebry.

Tedy pokud se omezíme na dvouhodnotovou logiku, potom relační operace pro zobecněný model splynou s těmi klasickými.

Při definici operací relační algebry pro zobecněný model často vycházíme z logického pohledu na klasické operace. Pomocí logické formule charakterizujeme výslednou relaci klasické relační operace. Tuto formuli poté interpretujeme ve fuzzy logice. Tento přístup je možné použít díky vztahům mezi klasickým a zobecněným doménovým relačním kalkulem a díky vzájemnému vztahu relační algebry a relačního kalkulu, které jsou blíže popsány v 3.4. (str. 30) a 3.5. (str. 37)

3.3.1. Množinové operace relační algebry

Klasická relační algebra disponuje operacemi, které nám umožňují vypočítat sjednocení, průnik či rozdíl dvou relací (nad stejným relačním schématem R). Klasické operace sjednocení a průniku lze definovat takto:

$$\mathcal{D}_1 \cup \mathcal{D}_2 = \{r \in \mathrm{Tupl}(R) \mid r \in \mathcal{D}_1 \text{ nebo } r \in \mathcal{D}_2\}$$
$$\mathcal{D}_1 \cap \mathcal{D}_2 = \{r \in \mathrm{Tupl}(R) \mid r \in \mathcal{D}_1 \text{ a } r \in \mathcal{D}_2\}$$

Nabízí se ovšem ještě alternativní pohled založený na kvantifikátorech:

$$\mathcal{D}_1 \cup \mathcal{D}_2 = \{r \in \mathrm{Tupl}(R) \mid \text{existuje relace } \mathcal{D} \in \{\mathcal{D}_1, \mathcal{D}_2\} \text{ tak, že } r \in \mathcal{D}\}$$
$$\mathcal{D}_1 \cap \mathcal{D}_2 = \{r \in \mathrm{Tupl}(R) \mid \text{pro všechny relace } \mathcal{D} \in \{\mathcal{D}_1, \mathcal{D}_2\} \text{ platí } r \in \mathcal{D}\}$$

Oba pohledy jsou v klasické logice ekvivalentní, nicméně v obecnější logice tomu tak již být nemusí. Proto v relační algebře pro zobecněný model uvažujeme operace vycházející z obou pohledů.

Jak bylo nastíněno v úvodu této kapitoly, postup odvození zobecněných operací je následující – vycházíme z formule charakterizující výsledek klasických operací, ve které explicitně uvažujeme stupně příslušnosti n-tic do odpovídajících relací a kterou interpretujeme v uvažované obecnější logice. Na symbolické úrovni se význam klasických a obecných operací neliší, tedy např. pokud relace $\mathcal{D}_1$ reprezentuje výsledek dotazu Q_1 a $\mathcal{D}_2$ výsledek dotazu Q_2, potom $(\mathcal{D}_1 \cup \mathcal{D}_2)(r)$ v klasickém i obecném případě interpretujeme jako stupeň, ve kterém n-tice r vyhovuje dotazu Q_1 *nebo* Q_2. [3] Avšak zatímco v klasickém případě může n-tice do relace pouze zcela patřit, nebo vůbec nepatřit, v obecném případě můžeme uvažovat i mezilehlé stupně.

V následujících definicích je první operace založena na standardní definici, druhá potom alternativní definici pomocí kvantifikátoru. Pro operace sjednocení relací s ranky $\mathcal{D}_1$ a $\mathcal{D}_2$ nad schématem R máme

$$(\mathcal{D}_1 \oplus \mathcal{D}_2)(r) = \mathcal{D}_1(r) \oplus \mathcal{D}_2(r) \tag{1}$$
$$(\mathcal{D}_1 \cup \mathcal{D}_2)(r) = \mathcal{D}_1(r) \vee \mathcal{D}_2(r) \tag{2}$$

pro všechny $r \in \mathrm{Tupl}(R)$. Operace neidempotentní disjunkce $\oplus$ se běžně neuvažuje, nicméně je možné ji do struktury pravdivostních hodnot přidat.

Relační operaci $\mathcal{D}_1 \oplus \mathcal{D}_2$ nazveme $\oplus$-sjednocení nebo neidempotentní sjednocení. Operaci $\mathcal{D}_1 \vee \mathcal{D}_2$ potom obdobně nazveme $\vee$-sjednocení nebo jen sjednocení [3], pokud $\oplus$-sjednocení neuvažujeme. Operace průniků relací pro všechny $r \in \mathrm{Tupl}(R)$ definujeme jako

$$(\mathcal{D}_1 \otimes \mathcal{D}_2)(r) = \mathcal{D}_1(r) \otimes \mathcal{D}_2(r) \tag{3}$$
$$(\mathcal{D}_1 \cap \mathcal{D}_2)(r) = \mathcal{D}_1(r) \wedge \mathcal{D}_2(r) \tag{4}$$

Operace nazýváme $\otimes$-, resp. $\wedge$-průnik. Operace $\otimes$-průniku je neidempotentní ($\mathcal{D} \otimes \mathcal{D} \subseteq \mathcal{D}$), $\wedge$-průnik je idempotentní ($\mathcal{D} \cap \mathcal{D} = \mathcal{D}$) [3].

Přítomnost idempotentních i neidempotentních operací v modelu je opodstatněná – operace se vůči rankům chovají rozdílným způsobem, přičemž každý ze způsobů je vhodnější pro jiný typ dotazů. Tabulka 3. ilustruje chování obou operací průniku. Rank n-tice v $\wedge$-průniku záleží vždy jen na nejhorším ranku n-tice (bez ohledu na ranky v ostatních relacích). Oproti tomu rank v $\otimes$-průniku relací je ovlivněn ranky ze všech vstupních relací. Protože n-tice z prvního řádku má skoro ve všech vstupních relacích vyšší rank než n-tice z druhého řádku, bude vyšší i její rank v $\otimes$-průniku. [3]

$\mathcal{D}_1(r)$	$\mathcal{D}_2(r)$	$\mathcal{D}_3(r)$	$\ldots$	$\mathcal{D}_k(r)$	$(\mathcal{D}_1 \cap \cdots \cap \mathcal{D}_k)(r)$	$(\mathcal{D}_1 \otimes \cdots \otimes \mathcal{D}_k)(r)$
0.5	0.98	0.98	$\ldots$	0.98	0.5	$0.5 \cdot 0.98^{k-1}$
0.5	0.5	0.5	$\ldots$	0.5	0.5	0.5^k

Tabulka 3.: Tabulka ilustruje chování operací $\otimes$- a $\wedge$-průniku. Strukturu pravdivostních hodnot uvažujeme na intervalu $[0, 1]$ s operacemi $a \wedge b = \min(a, b)$ a $a \otimes b = ab$. Příklad převzat z [3].

Dalšími operacemi, které klasická relační algebra poskytuje, jsou množinový rozdíl a (aktivní) doplněk. Pomocí těchto operací je možné vyjádřit dotazy zahrnující negaci. V zobecněné relační algebře je pro tento účel zavedena obecnější operace založená na operaci rezidua – dotazy s negací lze potom vyjádřit jako speciální případ této nové operace. [3]

Zavedení této operace není stejně přímočaré jako u předchozích operací průniku či sjednocení. Uvažujme relace $\mathcal{D}_1$ a $\mathcal{D}_2$ nad relačním schématem R a doménami D_y ($y \in R$). Pokud je alespoň jedna z těchto domén nekonečná, potom pro nekonečně mnoho n-tic $r \in \mathrm{Tupl}(R)$ platí $\mathcal{D}_1(r) \to \mathcal{D}_2(r) = 1$. Takto zavedená operace není doménově nezávislá a generuje nekonečné výsledné relace. [3]

Tento problém je vyřešen na úrovni struktury pravdivostních hodnot zavedením nové ternární operace *a-residua*, která je pro všechna $a, b, c \in L$ definována jako $b \twoheadrightarrow^a c = a \otimes (b \to c)$. Příslušná relační operace je potom pro relace $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ nad relačním schématem R definována jako [3]

$$(\mathcal{D}_1 \twoheadrightarrow^{\mathcal{D}_3} \mathcal{D}_2)(r) = \mathcal{D}_1(r) \twoheadrightarrow^{\mathcal{D}_3(r)} \mathcal{D}_2(r)$$

Operaci $\mathcal{D}_1 \twoheadrightarrow^{\mathcal{D}_3} \mathcal{D}_2$ nazýváme residuum $\mathcal{D}_1$ vzhledem k $\mathcal{D}_2$ nad $\mathcal{D}_3$. Na relaci $\mathcal{D}_3$ lze nahlížet jako na nové universum, nad kterým uvažujeme residuum. Platí totiž vztah $\mathcal{D}_1 \twoheadrightarrow^{\mathcal{D}_3} \mathcal{D}_2 \subseteq \mathcal{D}_3$. Snadno nahlédneme, že takto definovaná operace již netrpí problémem s nekonečnou výslednou relací. Pokud relace $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ představují výsledky dotazů Q_1, Q_2, Q_3, potom $(\mathcal{D}_1 \twoheadrightarrow^{\mathcal{D}_3} \mathcal{D}_2)(r)$ pro $r \in \mathrm{Tupl}(R)$ představuje stupeň, ve kterém r vyhovuje Q_3 *a pokud* vyhovuje Q_1, *potom* vyhovuje Q_2. [3]

Z definice residua vycházejí dvě třídy binárních relačních operací, kdy nahradíme jednu z relací konstantním pravdivostním stupněm. První operací je *a-negace* relace $\mathcal{D}_1$ nad $\mathcal{D}_2$ (obě relace uvažujeme nad relačním schématem R), kterou definujeme jako

$$(\mathcal{D}_2 \boxbslash_a \mathcal{D}_1)(r) = \mathcal{D}_1(r) \twoheadrightarrow^{\mathcal{D}_2(r)} a$$

pro všechny $r \in R$. Pokud relace $\mathcal{D}_1, \mathcal{D}_2$ představují výsledky dotazů Q_1, Q_2, potom $(\mathcal{D}_2 \boxbslash_a \mathcal{D}_1)(r)$ je stupeň, ve kterém r vyhovuje Q_2 a dotazu Q_1 vyhovuje nejvýše ve stupni a (speciálně pro $a = 0$ tedy interpretujeme jako r nevyhovuje dotazu Q_1). Pro $a = 0$ lze $\mathcal{D}_2 \boxbslash_0 \mathcal{D}_1$ chápat jako operaci množinového rozdílu. [3]

Operaci množinového rozdílu je alternativně možné definovat pomocí nezávislé logické operace abjunkce, kterou je nutné ve struktuře pravdivostních hodnot uvažovat jako základní operaci. Pro tento účel lze obohatit residuovaný svaz o dvojici operací neidempotentní disjunkce $\oplus$ a s ní provázanou operaci abjunkce (nebo též nonimplikace či rozdíl) $\ominus$ (nebo též $\nrightarrow$). Tato struktura se potom nazývá dvojitě residuovaný svaz [34]. [3]

Druhou operací založenou na residuu je *a-shift*, jehož definice je

$$(a \twoheadrightarrow^{\mathcal{D}_2} \mathcal{D}_1)(r) = a \twoheadrightarrow^{\mathcal{D}_2(r)} \mathcal{D}_1(r)$$

pro všechny $r \in \mathrm{Tupl}(R)$ a interpretujeme jej jako stupeň, ve kterém r splňuje dotaz Q_2 a Q_1 splňuje alespoň ve stupni a. [3]

3.3.2. Přirozené spojení (na rovnost)

Relační algebra pro zoběcněný relační model umožňuje uvažovat celou řadu různých operací spojení. Základní operací je však pouze spojení na rovnost, zbylé operace je možné odvodit. [3]

Nejprve zavedeme obvyklý pojem *spojitelnosti* n-tic. Řekneme, že n-tice r nad relačním schématem R a s nad S jsou spojitelné (v této práci ozn. $r \between s$), jestliže se shodují na společných atributech, tedy jestliže platí $r(y) = s(y)$ pro $y \in R \cap S$. Pokud jsou n-tice r a s spojitelné, uvažujeme jejich *spojení*. Spojením n-tic r a s rozumíme n-tici rs nad relačním schématem RS[2] takovou, že $rs(y) = r(y)$ pro $y \in R$ a $rs(y) = s(y)$ pro $y \in S$. [3]

[2]Pro relační schémata X a Y zápisem XY rozumíme jejich sjednocení $X \cup Y$.

Nyní uvažujme relace $\mathcal{D}_1$ nad relačním schématem RS a $\mathcal{D}_2$ nad ST, kde R, S, T jsou po dvou disjunktní ($R \cap S = R \cap T = S \cap T = \emptyset$). V klasické relační algebře je spojení dáno předpisem

$$\mathcal{D}_1 \bowtie \mathcal{D}_2 = \{rst \in \mathrm{Tupl}(RST) \mid rs \in \mathcal{D}_1 \text{ a } st \in \mathcal{D}_2\}.$$

Přímočarým zobecněním této definice obdržíme operaci spojení (na rovnost) v zobecněné relační algebře. Výsledná relace této operace je nad schématem RST a je dána předpisem

$$(\mathcal{D}_1 \bowtie \mathcal{D}_2)(rst) = \mathcal{D}_1(rs) \otimes \mathcal{D}_2(st)$$

pro každé $r \in \mathrm{Tupl}(R), s \in \mathrm{Tupl}(S)$ a $t \in \mathrm{Tupl}(S)$. Pro $S = \emptyset$ operaci nazveme kartézský součin. Pokud relace $\mathcal{D}_1$ a $\mathcal{D}_2$ reprezentují výsledky dotazů Q_1 a Q_2, potom rank $(\mathcal{D}_1 \bowtie \mathcal{D}_2)(rst)$ odpovídá stupni, ve kterém n-tice rs vyhovuje dotazu Q_1 *a* n-tice st dotazu Q_2. Tato interpretace je ve shodě s tím, jak spojení chápeme v klasické relační algebře. [3]

Pomocí spojení je možné vyjádřit selekci na rovnost atributu a hodnoty. K tomuto účelu využijeme tzv. singleton $[y\colon d]$, což je relace nad relačním schématem $\{y\}$ s jedinou n-ticí t, pro kterou platí $t(y) = d$. Pro selekci na rovnost máme [3]

$$\sigma_{y=\mathfrak{d}}(\mathcal{D})(r) = (\mathcal{D} \bowtie [y\colon d])(r) = \begin{cases} \mathcal{D}(r) & \text{pokud } r(y) = d, \\ 0 & \text{jinak.} \end{cases}$$

pro každou n-tici $r \in \mathrm{Tupl}(R)$.

3.3.3. Projekce a dělení

Relační algebra dále poskytuje operace, které umožňují vyjadřovat dotazy zahrnující kvantifikátory. Těmito operacemi jsou projekce pro dotazy s existenční kvantifikací a dělení pro dotazy s všeobecnou kvantifikací.

Relační dělení je nutné zahrnout do relační algebry zobecněného modelu jako základní operaci. Zatímco v klasické relační algebře je možné relační dělení definovat na základě vzájemného vztahu kvantifikátorů platného v klasické logice, v obecnější logice tento vztah již neplatí. Pokud bychom operaci relačního dělení do zobecněné relační algebry nezahrnuli, nebyli bychom schopni vyjádřit dotazy s všeobecnou kvantifikací a nebylo by možné prokázat relační úplnost jazyka založeného na relační algebře vzhledem k relačnímu kalkulu. [3]

Před definicí operace projekce nejprve zavedeme standardní pojem zúžení (projekce) n-tice. Mějme n-tici t nad relačním schématem T, zúžením n-tice t na relační schéma $R \subseteq T$ rozumíme n-tici $t(R)$ nad R takovou, že $t(R)(y) = t(y)$ pro všechna $y \in R$. Nyní uvažujme relaci $\mathcal{D}$ nad relačním schématem T a relační schéma $R \subseteq T$. V klasické relační algebře lze na projekci nahlížet takto

$$\pi_R(\mathcal{D}) = \{r \in \mathrm{Tupl}(R) \mid \text{existuje } s \in \mathrm{Tupl}(T \setminus R) \text{ tak, že } rs \in \mathcal{D}\}.$$

Je možný i alternativní pohled založený na zúžení n-tice

$$\pi_R(\mathcal{D}) = \{r \in \mathrm{Tupl}(R) \mid \text{existuje } t \in \mathcal{D} \text{ takové, že } r = t(R)\}.$$

Oba pohledy jsou rovnocenné a vedou na následující definici zobecněné operace projekce. Pro všechny n-tice $r \in \mathrm{Tupl}(R)$ máme [3]

$$(\pi_R(\mathcal{D}))(r) = \bigvee_{s \in \mathrm{Tupl}(T \setminus R)} \mathcal{D}(rs) \qquad (\text{alternativně} \bigvee_{\substack{t \in \mathrm{Tupl}(T) \\ r = t(R)}} \mathcal{D}(t)).$$

Nyní přesuneme pozornost na operaci relačního dělení. Jelikož je této operaci v práci věnována samostatná kapitola 4. (str. 38), zde pouze pro úplnost stručně nastíníme definici relačního dělení podle [3], která v jistém ohledu vychází z Coddova relačního dělení. Aby byla zajištěna konečnost výsledné relace, je operace založena na ternární operaci *a-residua*, podobně jako relační operace residua.

Uvažujme relace $\mathcal{D}_1$ nad relačním schématem R, $\mathcal{D}_2$ nad $S \subseteq R$ a $\mathcal{D}_3$ nad $T = R \setminus S$. Výsledek operace relačního dělení $\mathcal{D}_1 \div^{\mathcal{D}_3} \mathcal{D}_2$ (podíl relací $\mathcal{D}_1$ a $\mathcal{D}_2$ nad universem $\mathcal{D}_3$) je nad relačním schématem T a definujeme jej jako [3]

$$(\mathcal{D}_1 \div^{\mathcal{D}_3} \mathcal{D}_2)(t) = \bigwedge_{s \in \mathrm{Tupl}(S)} (\mathcal{D}_2(s) \twoheadrightarrow^{\mathcal{D}_3(t)} \mathcal{D}_1(st))$$

pro každou n-tici $t \in \mathrm{Tupl}(T)$. Pokud relace $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ reprezentují výsledky dotazů Q_1, Q_2, Q_3, potom $(\mathcal{D}_1 \div^{\mathcal{D}_3} \mathcal{D}_2)(t)$ interpretujeme jako stupeň, ve kterém n-tice $t \in \mathrm{Tupl}(T)$ vyhovuje Q_3 a *pro každou* n-tici $s \in \mathrm{Tupl}(S)$, která vyhovuje Q_2, n-tice st vyhovuje Q_1.

Pokud máme dvě relace nad stejným relačním schématem (např. R) a namísto $\mathcal{D}_3$ použijeme ve výše uvedené definici dělení relaci $1_\emptyset$ reprezentující pravdivostní hodnotu 1, dostaneme operaci vyjadřují vztah „býti podmnožinou ve stupni“. Výsledek této operace je nad prázdným relačním schématem, je vždy roven některé z relací $a_\emptyset$ a je ve tvaru [3]

$$S(\mathcal{D}_1, \mathcal{D}_2) = (\mathcal{D}_2 \div^{1_\emptyset} \mathcal{D}_1)(\emptyset) = \bigwedge_{r \in \mathrm{Tupl}(R)} (\mathcal{D}_1(r) \to \mathcal{D}_2(r)).$$

3.3.4. Podobnostní selekce, kernel a support, přejmenování atributů

Operace selekce umožňuje z příslušné relace vybrat n-tice, které vyhovují námi zadané podmínce. V klasické relační algebře základní varianta operace selekce umožňuje specifikovat podmínky ve tvaru rovnosti dvou atributů ($x = y$) nebo rovnosti atributu a konstanty ($x = d$). Zobecněná operace selekce namísto rovností využívá podobností, a tím nám poskytuje prostředky pro vyjádření podobnostních podmínek a dotazů, např. „x přibližně rovno y“ ($x \approx y$).

Vyžadujeme, aby atributy a objektové konstanty vstupující do podobnostních podmínek byly stejného typu. Díky tomu máme zajištěno, že uvažujeme atributy

a konstanty nad stejnou doménou se stejnou relací podobnosti a námi zadaná podmínka má smysl. Z požadavku na to, aby domény $\{\langle D_y, \approx_y\rangle \mid y \in Y\}$ *respektovaly* typovou deklaraci $\mathbf{\Lambda} = \langle \Lambda, \lambda\rangle$, totiž pro atributy $x, y \in Y$ stejného typu $(\lambda(x) = \lambda(y))$ plyne, že jejich příslušné domény $\langle D_x, \approx_x\rangle$ a $\langle D_y, \approx_y\rangle$ včetně relací podobností splývají (obdobně potom pro atribut a konstantu). [3]

Pro relaci $\mathcal{D}$ nad relačním schématem R a pro atributy $y_1, y_2 \in R$ stejného typu $(\lambda(y_1) = \lambda(y_2))$ definujeme výsledek podobnostní selekce nad R takto

$$(\sigma_{y_1 \approx y_2}(\mathcal{D}))(r) = \mathcal{D}(r) \otimes r(y_1) \approx_{y_1} r(y_2)$$

pro všechny $r \in \mathrm{Tupl}(R)$. Pokud relace $\mathcal{D}$ reprezentuje výsledek dotazu Q, potom $(\sigma_{y_1 \approx y_2}(\mathcal{D}))(r)$ představuje stupeň, ve kterém r vyhovuje dotazu Q a současně je jeho hodnota atributu y_1 podobná hodnotě atributu y_2 v r. [3]

Obdobně můžeme operaci selekce zavést pro atribut y a hodnotu $d \in D_y$ jako

$$(\sigma_{y \approx d}(\mathcal{D}))(r) = \mathcal{D}(r) \otimes r(y) \approx_y d$$

pro každou n-tici $r \in \mathrm{Tupl}(R)$. Tuto operaci je však možné odvodit pomocí spojení, singletonu a selekce na podobnost dvou atributů. Vezmeme atribut $y' \notin R$ stejného typu jako y $(\lambda(y) = \lambda(y'))$ a selekci na podobnost atributu a hodnoty můžeme vyjádřit jako [3]

$$\sigma_{y \approx d}(\mathcal{D}) = \pi_R(\sigma_{y \approx y'}(\mathcal{D} \bowtie [y' \colon d])).$$

Zajímavými operacemi, které nemají v klasické relační algebře netriviální protějšky, jsou kernel a support. Tyto operace nám umožňují transformovat relaci s obecnými ranky na relaci, ve které má každá n-tice rank 0 nebo 1 (tedy na relace odpovídající klasickým). Díky těmto operacím je možné z podobnostních operací vytvořit klasické operace (např. z podobnostní selekce vytvořit selekci na rovnost apod.). [3]

Pro relaci $\mathcal{D}$ nad relačním schématem R výsledky operací kernel $\Delta(\mathcal{D})$ a support $\nabla(\mathcal{D})$ definujeme jako relace nad R ve tvaru:

$$(\Delta(\mathcal{D}))(r) = \begin{cases} 1 & \text{pokud } \mathcal{D}(r) = 1, \\ 0 & \text{jinak,} \end{cases}$$

$$(\nabla(\mathcal{D}))(r) = \begin{cases} 1 & \text{pokud } \mathcal{D}(r) > 0, \\ 0 & \text{jinak,} \end{cases}$$

pro každou n-tici $r \in \mathrm{Tupl}(R)$. Pokud relace $\mathcal{D}$ reprezentuje výsledek dotazu Q, potom kernel $\Delta(\mathcal{D})$ odpovídá relaci, jež obsahuje pouze ty n-tice, které plně vyhovují Q. Relace support $\nabla(\mathcal{D})$ potom zahrnuje n-tice, které alespoň částečně vyhovují Q. [3]

Stejně jako v klasickém modelu uvažujeme přejmenování atributů ρ_h. Vyžadujeme, aby nové atributy dané injektivním zobrazením $h \colon R \to Y$ měly stejný typ jako jejich vzory. Přesná definice přejmenování je uvedena v [3].

3.3.5. Dotazovací jazyk založený na relační algebře

Na relační algebře je přímo založen jeden ze základních dotazovacích jazyků, ve kterém dotazy odpovídají postupné aplikaci relačních operací na relace. Formálně, dotazem rozumíme tzv. *výraz relační algebry* (zkráceně RA-výraz). Odpověď na dotaz následně získáme vyhodnocením RA-výrazu v instanci databáze. [3]

Výrazy relační algebry (RA-výrazy) pro databázové schéma $\langle \mathbb{R}, \varrho \rangle$ nad Y a typovou deklaraci $\mathbf{\Lambda} = \langle \Lambda, \lambda \rangle$ pro Y a $\mathfrak{C}$ definujeme rekurzivně takto: [3]

1. Pro $\mathbb{r} \in \mathbb{R}$ je $\mathbb{r}$ RA-výraz nad relačním schématem $\varrho(\mathbb{r})$,
2. pro $a \in L$ je $\bar{a}_\emptyset$ RA-výraz nad relačním schématem $\emptyset$,
3. pro $\mathfrak{d} \in \mathfrak{C}$ a $y \in Y$, pokud platí $\lambda(\mathfrak{d}) = \lambda(y)$, potom $[y\colon \mathfrak{d}]$ je RA-výraz nad relačním schématem $\{y\}$,
4. pokud jsou E_1 a E_2 RA-výrazy nad relačním schématem R, potom $(E_1 \cup E_2)$ a $(E_1 \cap E_2)$ jsou RA-výrazy nad R,
5. pokud jsou E_1, E_2 a E_3 RA-výrazy nad relačním schématem R, potom $(E_1 \twoheadrightarrow^{E_3} E_2)$ je RA-výraz nad R,
6. pokud je E_1 RA-výraz nad R_1 a E_2 RA-výraz nad R_2, potom $(E_1 \bowtie E_2)$ je RA-výraz nad $R_1 R_2$,
7. pokud je E RA-výraz nad T a máme $R \subseteq T$, potom $\pi_R(E)$ je RA-výraz nad R,
8. pokud je E_1 RA-výraz nad R, E_2 RA-výraz nad $S \subseteq R$ a E_3 je RA-výraz nad $T = R \setminus S$, potom $(E_1 \div^{E_3} E_2)$ je RA-výraz nad T,
9. pokud je E RA-výraz nad R, $y \in R$ a $z \in R \cup \mathfrak{C}$ takové, že $\lambda(y) = \lambda(z)$, potom $\sigma_{y \approx z}(E)$ je RA-výraz nad R,
10. pokud je E RA-výraz nad R, potom $\Delta(E)$ a $\nabla(E)$ jsou RA-výrazy nad R,
11. pokud je E RA-výraz nad R a $h\colon R \to Y$ injektivní zobrazení takové, že pro každé $y \in R$ máme $\lambda(y) = \lambda(h(y))$, potom $\rho_h(E)$ je RA-výraz nad $h(R)$.

Každý RA-výraz E má své relační schéma, které označujeme jako $\mathrm{sch}(E)$. Z logického pohledu je možné RA-výrazy považovat za termy predikátové logiky, jejíž jazyk je dán databázovým schématem $\langle \mathbb{R}, \varrho \rangle$, množinou objektových konstant $\mathfrak{C}$, typovou deklarací $\mathbf{\Lambda}$ a množinou pravdivostních hodnot L. [3]

RA-výrazy jsou čistě syntaktické pojmy a samy o sobě nemají žádnou hodnotu. Aby bylo možné dotazy ve formě RA-výrazů vyhodnocovat, musíme mít k dispozici vhodnou strukturu, pomocí které jednotlivým RA-výrazům přiřadíme

jejich hodnotu. Protože chceme dotazy pokládat v námi uvažované databázi, roli vhodné struktury hraje instance této databáze. Pokud je výsledek RA-výrazu E definován, pak se jedná o relaci s ranky nad relačním schématem $\mathrm{sch}(E)$. [3]

Uvažujme instanci databáze $\boldsymbol{\mathcal{D}} = \langle U^{\boldsymbol{\mathcal{D}}}, \mathbb{R}^{\boldsymbol{\mathcal{D}}}, \mathfrak{C}^{\boldsymbol{\mathcal{D}}} \rangle$ nad databázovým schématem $\langle \mathbb{R}, \varrho \rangle$, pro jejíž domény z $U^{\boldsymbol{\mathcal{D}}}$ platí, že respektují typovou deklaraci $\boldsymbol{\Lambda} = \langle \Lambda, \lambda \rangle$ pro Y a $\mathfrak{C}$. Potom pro RA-výrazy E nad $\langle \mathbb{R}, \varrho \rangle$ definujeme relaci $E^{\boldsymbol{\mathcal{D}}}$, kterou nazýváme výsledek E v $\boldsymbol{\mathcal{D}}$, takto: [3]

1.	Pro E ve tvaru	$\mathbb{r} \in \mathbb{R}$	položíme $E^{\boldsymbol{\mathcal{D}}} = \mathbb{r}^{\boldsymbol{\mathcal{D}}}$,
2.	pro E ve tvaru	$\bar{a}_{\emptyset}$	položíme $E^{\boldsymbol{\mathcal{D}}} = a_{\emptyset}$,
3.	pro E ve tvaru	$[y\colon \mathfrak{d}]$	položíme $E^{\boldsymbol{\mathcal{D}}} = [y\colon \mathfrak{d}^{\boldsymbol{\mathcal{D}}}]$,
4.	pro E ve tvaru	$E_1 \cup E_2$	položíme $E^{\boldsymbol{\mathcal{D}}} = E_1^{\boldsymbol{\mathcal{D}}} \cup E_2^{\boldsymbol{\mathcal{D}}}$,
	pro E ve tvaru	$E_1 \cap E_2$	položíme $E^{\boldsymbol{\mathcal{D}}} = E_1^{\boldsymbol{\mathcal{D}}} \cap E_2^{\boldsymbol{\mathcal{D}}}$,
5.	pro E ve tvaru	$E_1 \twoheadrightarrow^{E_3} E_2$	položíme $E^{\boldsymbol{\mathcal{D}}} = E_1^{\boldsymbol{\mathcal{D}}} \twoheadrightarrow^{E_3^{\boldsymbol{\mathcal{D}}}} E_2^{\boldsymbol{\mathcal{D}}}$,
6.	pro E ve tvaru	$E_1 \bowtie E_2$	položíme $E^{\boldsymbol{\mathcal{D}}} = E_1^{\boldsymbol{\mathcal{D}}} \bowtie E_2^{\boldsymbol{\mathcal{D}}}$,
7.	pro E ve tvaru	$\pi_R(F)$	položíme $E^{\boldsymbol{\mathcal{D}}} = \pi_R(F^{\boldsymbol{\mathcal{D}}})$,
8.	pro E ve tvaru	$E_1 \div^{E_3} E_2$	položíme $E^{\boldsymbol{\mathcal{D}}} = E_1^{\boldsymbol{\mathcal{D}}} \div^{E_3^{\boldsymbol{\mathcal{D}}}} E_2^{\boldsymbol{\mathcal{D}}}$,
9.	pro E ve tvaru	$\sigma_{y \approx z}(F)$ a $z \in Y$	položíme $E^{\boldsymbol{\mathcal{D}}} = \sigma_{y \approx z}(F^{\boldsymbol{\mathcal{D}}})$,
	pro E ve tvaru	$\sigma_{y \approx \mathfrak{d}}(F)$ a $\mathfrak{d} \in \mathfrak{C}$	položíme $E^{\boldsymbol{\mathcal{D}}} = \sigma_{y \approx \mathfrak{d}^{\boldsymbol{\mathcal{D}}}}(F^{\boldsymbol{\mathcal{D}}})$,
10.	pro E ve tvaru	$\Delta(F)$	položíme $E^{\boldsymbol{\mathcal{D}}} = \Delta(F^{\boldsymbol{\mathcal{D}}})$,
	pro E ve tvaru	$\nabla(F)$	položíme $E^{\boldsymbol{\mathcal{D}}} = \nabla(F^{\boldsymbol{\mathcal{D}}})$,
11.	pro E ve tvaru	$\rho_h(F)$	položíme $E^{\boldsymbol{\mathcal{D}}} = \rho_h(F^{\boldsymbol{\mathcal{D}}})$.

Pomocí RA-výrazů singletonu $[y\colon \mathfrak{d}]$, $\bar{a}_{\emptyset}$, sjednocení a spojení je navíc možné vytvořit libovolnou konstantní relaci (skutečné hodnoty v této relaci jsou dány interpretací objektových konstant ve zvolené instanci databáze). Postup je následující. Nejprve zkonstruujeme příslušné n-tice nad uvažovaným relačním schématem $R = \{y_1, \cdots, y_n\}$ pomocí RA-výrazu $\bar{a}_{\emptyset} \bowtie (\bowtie_{i=1}^{n} [y_i\colon \mathfrak{d}_i])$. Konečně mnoha sjednoceními takových n-tic získáme RA-výraz, jehož vyhodnocením v instanci databáze dostaneme požadovanou relaci. [3]

3.4. Doménový relační kalkul

Druhým základním dotazovacím jazykem je relační kalkul. Zatímco v jazyce založeném na relační algebře dotaz odpovídá posloupnosti kroků, které je nutné pro výpočet odpovědi vykonat, v relačním kalkulu dotazem charakterizujeme vlastnosti, které musejí n-tice splňovat, aby mohly být zahrnuty ve výsledku dotazu.

Relační kalkul je založen na predikátové logice prvního řádu. Požadované vlastnosti specifikujeme pomocí formulí této logiky. Při vyhodnocování dotazu ohodnocujeme volné proměnné formule a pokud je tato formule pod daným ohodnocením pravdivá (obecně v nějakém stupni), můžeme n-tici odpovídající tomuto ohodnocení vložit do výsledné relace (obecně ve stupni, ve kterém byla formule pod tímto ohodnocením pravdivá).

Existují dvě varianty relačního kalkulu s ekvivalentní vyjadřovací silou – n-ticový a doménový relační kalkul. Kalkuly se liší významem objektových proměnných, kdy v prvně jmenovaném kalkulu objektové proměnné představují n-tice, v druhém potom zastupují hodnoty z domén atributů. [32] [39] Dále se budeme věnovat pouze doménovému relačnímu kalkulu a jeho zobecnění, jež je popsáno v [3].

Zobecněný doménový relační kalkul s deklaracemi rozsahu proměnných vychází z klasického doménového relačního kalkulu, který zavedli Lacroix a Pirotte v [29]. Základ tvoří formule predikátové logiky prvního řádu. Ačkoliv se kalkuly liší v použité logice, kdy v zobecněném kalkulu je namísto klasické dvouhodnotové logiky použita fuzzy logika, na syntaktické úrovni jsou oba kalkuly totožné. Díky tomu formule klasického kalkulu přenesené do zobecněného kalkulu vyjadřují stejný koncept, jen jeho splnění uvažujeme ve stupních. Tento fakt představuje jednu z nejvýznamnějších vlastností zobecněného relačního modelu, která nám umožňuje snadné zobecňování pojmů z klasického modelu. [3]

V dalších částech popíšeme základní pojmy, syntaxi a sémantiku doménového relačního kalkulu. Jazyk predikátové logiky, jenž budeme v dalších částech potřebovat a nad kterým budeme uvažovat formule, je dán databázovým schématem $\langle \mathbb{R}, \varrho \rangle$, množinou objektových konstant $\mathfrak{C}$, konstantami pro pravdivostní hodnoty (pro každé $a \in L$ uvažujeme $\bar{a}$). Jazyk dále obsahuje spočetně mnoho objektových proměnných, symboly pro logické spojky a kvantifikátory a pomocné symboly (závorky, čárka, dvojtečka). [3]

3.4.1. Proměnné, deklarace rozsahu

Předpokládáme, že máme k dispozici spočetně mnoho objektových proměnných $\mathbb{x}, \mathbb{x}_1, \mathbb{x}_2, \ldots \in X$, které zastupují hodnoty z domén. Narozdíl od objektových konstant, jejichž interpretace je pevně dána instancí databáze, objektové proměnné mohou nabývat libovolné hodnoty z libovolné domény. Abychom zajistili konečnost výsledků dotazů v relačním kalkulu, zavádíme

deklaraci rozsahu proměnných. Tato deklarace, interpretovaná v instanci databáze, každé objektové proměnné přiřadí konečnou množinu hodnot, kterých může daná proměnná nabývat. [3]

Formálně, deklarací rozsahu R nad $\langle \mathbb{R}, \varrho \rangle$ a $\mathfrak{C}$ rozumíme konečnou neprázdnou množinu tvořenou *částmi rozsahu*, kde částí rozsahu rozumíme libovolnou objektovou konstantu z $\mathfrak{C}$ anebo výraz $\mathfrak{r}(y)$, kde $\mathfrak{r} \in \mathbb{R}$ a $y \in \varrho(\mathfrak{r})$ (tedy $\mathfrak{r}(y)$ označuje konkrétní atribut y v relačním schématu relačního symbolu $\mathfrak{r}$). Řekneme, že deklarace rozsahu je kompatibilní s typovou deklarací $\mathbf{\Lambda} = \langle \Lambda, \lambda \rangle$, jestliže všechny části rozsahu jsou stejného typu $\tau \in \Lambda$. Tento typ τ nazveme typem deklarace rozsahu R a označujeme jej $\lambda(\mathsf{R})$. Množinu všech deklarací rozsahu nad $\langle \mathbb{R}, \varrho \rangle$ a $\mathfrak{C}$ kompatibilních s λ označujeme jako $\mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \mathbf{\Lambda})$. Zobrazení $rd\colon X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \mathbf{\Lambda})$ nazveme jako deklarace rozsahu pro objektové proměnné z X. Hodnotou $rd(\mathfrak{r})$ potom rozumíme deklaraci rozsahu pro $\mathfrak{r}$. [3]

Při vyhodnocování dotazu v instanci databáze $\boldsymbol{\mathcal{D}} = \langle U^{\boldsymbol{\mathcal{D}}}, \mathbb{R}^{\boldsymbol{\mathcal{D}}}, \mathfrak{C}^{\boldsymbol{\mathcal{D}}} \rangle$ (nad databázovým schématem $\langle \mathbb{R}, \varrho \rangle$) potřebujeme znát skutečné hodnoty, které můžeme podle deklarace rozsahu $\mathsf{R} \in \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \mathbf{\Lambda})$ příslušným proměnným přiřadit. K tomuto účelu definujeme množinu hodnot $\mathsf{R}^{\boldsymbol{\mathcal{D}}}$ předpisem

$$\mathsf{R}^{\boldsymbol{\mathcal{D}}} = \{\mathfrak{c}^{\boldsymbol{\mathcal{D}}} \mid \mathfrak{c} \in \mathsf{R}\} \cup \{d \in D_y \mid \mathfrak{r}(y) \in \mathsf{R} \text{ a } \left(\pi_{\{y\}}(\mathfrak{r}^{\boldsymbol{\mathcal{D}}})\right)(\{\langle y, d \rangle\}) > 0\}$$

a nazýváme ji hodnota R v instanci databáze $\boldsymbol{\mathcal{D}}$ [3]. Množina $\mathsf{R}^{\boldsymbol{\mathcal{D}}}$ je tedy na jedné straně tvořena hodnotami, kterými v dané instanci databáze interpretujeme objektové konstanty, a na druhé straně pak hodnotami, které se nacházejí v odpovídajícím „sloupci“ relace, pomocí které interpretujeme příslušný relační symbol.

Pro každou množinu $\mathsf{R}^{\boldsymbol{\mathcal{D}}}$ navíc existuje RA-výraz, jehož vyhodnocením v instanci databáze získáme relaci nad jednoprvkovým relačním schématem obsahující n-tice, kde hodnoty jejich jediného atributu přesně odpovídají hodnotám z $\mathsf{R}^{\boldsymbol{\mathcal{D}}}$. Přesněji, uvažujme deklaraci rozsahu R kompatibilní s typovou deklarací $\mathbf{\Lambda} = \langle \Lambda, \lambda \rangle$. Pro každý atribut y, jehož typ odpovídá typu deklarace rozsahu ($\lambda(y) = \lambda(\mathsf{R})$), existuje RA-výraz $E_{\mathsf{R},y}$ nad relačním schématem $\{y\}$ takový, že pro libovolnou n-tici $r \in \mathrm{Tupl}(\{y\})$ a instanci databáze $\boldsymbol{\mathcal{D}}$ platí

$$\left(E^{\boldsymbol{\mathcal{D}}}_{\mathsf{R},y}\right)(r) = \begin{cases} 1 & \text{pokud } r(y) \in \mathsf{R}^{\boldsymbol{\mathcal{D}}}, \\ 0 & \text{jinak.} \end{cases}$$

Tento RA-výraz je ve tvaru [3]:

$$E_{\mathsf{R},y} = \bigcup_{\mathfrak{c} \in \mathsf{R}} ([y\colon \mathfrak{c}]) \cup \bigcup_{\mathfrak{r}(z) \in \mathsf{R}} \left(\varrho_{y \leftarrow z}(\pi_{\{z\}}(\nabla(\mathfrak{r})))\right).$$

3.4.2. Formule doménového relačního kalkulu

Základní stavební prvek dotazů doménového relačního kalkulu tvoří formule. Pomocí formulí popisujeme vlastnosti, které musejí n-tice splňovat, aby mohly

být zařazeny do odpovědi na daný dotaz. Formule doménového relačního kalkulu definujeme následujícím způsobem.

Mějme databázové schéma $\langle \mathbb{R}, \varrho \rangle$ nad Y, typovou deklaraci $\mathbf{\Lambda}$ pro Y a $\mathfrak{C}$ a množinu objektových proměnných X. Formulemi doménového relačního kalkulu nad $\langle \mathbb{R}, \varrho \rangle, \mathfrak{C}$ a $\mathbf{L}$ jsou [3]:

1. Pro relační symbol $\mathbb{r} \in \mathbb{R}$ s relačním schématem $\varrho(\mathbb{r}) = \{y_1, \ldots, y_n\}$ a pro objektové proměnné $\mathbb{x}_1, \ldots, \mathbb{x}_n \in X$ je $\mathbb{r}(y_1 \colon \mathbb{x}_1, \ldots, y_n \colon \mathbb{x}_n)$ formule[3],
2. pro $a \in L$ je $\bar{a}$ formule,
3. pro $\mathbb{x} \in X$ a $z \in X \cup \mathfrak{C}$ je $\mathbb{x} \approx \mathbb{z}$ formule,
4. pokud jsou φ a ψ formule, potom $(\varphi \otimes \psi)$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$ a $(\varphi \Rightarrow \psi)$ jsou formule,
5. pokud je φ formule, potom $\Delta\varphi$ je formule,
6. pokud je φ formule, $\mathbb{x} \in X$ objektová proměnná a $\mathsf{R} \in \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \mathbf{\Lambda})$ deklarace rozsahu, potom $(\forall \mathbb{x} \in \mathsf{R})\varphi$ a $(\exists \mathbb{x} \in \mathsf{R})\varphi$ jsou formule.

Aby nebylo možné vyhodnocením dotazu vygenerovat nekonečnou relaci, byla zavedena deklarace rozsahu pro objektové proměnné, která omezuje množinu přípustných hodnot, kterých mohou tyto proměnné nabývat. Obecné formule však deklaraci rozsahu neberou v potaz, navíc je možné zkonstruovat formule, které z hlediska typové deklarace nedávají smysl. Z těchto důvodů se v dotazech omezujeme pouze na *bezpečné* formule, které těmito problémy netrpí. Řekneme, že formule φ je bezpečná vzhledem k $rd : X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \mathbf{\Lambda})$, pokud pro každou proměnnou $\mathbb{x}$, která se vyskytuje ve φ, platí [3]:

- Pokud má $\mathbb{x}$ v podformuli ψ formule φ volný výskyt a
  - ψ je ve tvaru $\mathbb{r}(y \colon \mathbb{x}, \ldots)$, potom atribut y a rozsah platnosti pro $\mathbb{x}$ musí být stejného typu, tedy $\lambda(y) = \lambda(rd(\mathbb{x}))$, nebo
  - ψ je ve tvaru $\mathbb{x} \approx \mathfrak{c}$, potom musí platit $\lambda(\mathfrak{c}) = \lambda(rd(\mathbb{x}))$, nebo
  - ψ je ve tvaru $\mathbb{x} \approx \mathbb{y}$ a tento výskyt $\mathbb{y}$ je ve ψ volný, potom musí platit $\lambda(rd(\mathbb{x})) = \lambda(rd(\mathbb{y}))$.
- Pro každou podformuli formule φ, která je ve tvaru $(\forall \mathbb{x} \in \mathsf{R})\psi$ nebo $(\exists \mathbb{x} \in \mathsf{R})\psi$, musí platit, že ψ je bezpečná vzhledem k $rd' \colon X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \mathbf{\Lambda})$, kde pro každé $\mathbb{y} \neq \mathbb{x}$ máme $rd'(\mathbb{y}) = rd(\mathbb{y})$ a $rd'(\mathbb{x}) = \mathsf{R}$.

[3] Na pořadí proměnných v $\mathbb{r}(y_1 \colon \mathbb{x}_1, \ldots, y_n \colon \mathbb{x}_n)$ nezáleží.

3.4.3. Sémantika formulí doménového relačního kalkulu

Abychom se mohli zabývat pravdivostními hodnotami formulí, potřebujeme vhodnou strukturu pro jazyk formulí a příslušné ohodnocení volných proměnných.

Hledanou strukturou je instance databáze $\boldsymbol{\mathcal{D}} = \langle U^{\boldsymbol{\mathcal{D}}}, \mathbb{R}^{\boldsymbol{\mathcal{D}}}, \mathfrak{C}^{\boldsymbol{\mathcal{D}}} \rangle$ nad databázovým schématem $\langle \mathbb{R}, \varrho \rangle$ s doménami $U^{\boldsymbol{\mathcal{D}}} = \{\langle D_y, \approx_y \rangle \mid y \in Y\}$. Pro instanci databáze $\boldsymbol{\mathcal{D}}$ a deklaraci rozsahu $rd\colon X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \boldsymbol{\Lambda})$ definujeme $\boldsymbol{\mathcal{D}}$-ohodnocení proměnných vzhledem k deklaraci rozsahu rd pro proměnné jako zobrazení $\boldsymbol{v}\colon X \to \bigcup_{y \in Y} D_y$, pro které musí platit, že objektovým proměnným přiřazuje pouze hodnoty z příslušné množiny přípustných hodnot, která je dána deklarací rozsahu. Pro každé $\mathbb{x} \in X$ tedy musí platit $\boldsymbol{v}(\mathbb{x}) \in rd(\mathbb{x})^{\boldsymbol{\mathcal{D}}}$. Pokud se dvě $\boldsymbol{\mathcal{D}}$-ohodnocení $\boldsymbol{v}$ a $\boldsymbol{w}$ liší pouze v hodnotě, kterou přiřazují objektové proměnné $\mathbb{x}$, potom tento fakt zapíšeme jako $\boldsymbol{v} =_{\mathbb{x}} \boldsymbol{w}$. [3]

Nyní již můžeme definovat pravdivostní hodnotu formule při daném ohodnocení. Uvažujme instanci databáze $\boldsymbol{\mathcal{D}} = \langle U^{\boldsymbol{\mathcal{D}}}, \mathbb{R}^{\boldsymbol{\mathcal{D}}}, \mathfrak{C}^{\boldsymbol{\mathcal{D}}} \rangle$ nad $\langle \mathbb{R}, \varrho \rangle$, jejíž domény respektují typovou deklaraci $\boldsymbol{\Lambda} = \langle \Lambda, \lambda \rangle$ pro Y a $\mathfrak{C}$. Dále mějme $\boldsymbol{\mathcal{D}}$-ohodnocení vzhledem k $rd\colon X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \boldsymbol{\Lambda})$ a formuli φ, jež je bezpečná vzhledem k rd. Potom stupeň $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} \in L$, ve kterém je formule φ pravdivá v $\boldsymbol{\mathcal{D}}$ při $\boldsymbol{v}$ a rd, je definován takto: [3]

1. Pro φ ve tvaru $\mathbb{r}(y_1\colon \mathbb{x}_1, \ldots, y_n\colon \mathbb{x}_n)$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \mathbb{r}^{\boldsymbol{\mathcal{D}}}(t)$, kde t je n-tice nad relačním schématem $\varrho(\mathbb{r})$, jejíž hodnoty atributů y_i odpovídají ohodnocení příslušných proměnných $\mathbb{x}_i$, tedy pro každé $i = 1, \ldots, n$ platí $t(y_i) = \boldsymbol{v}(\mathbb{x}_i)$,

2. pro φ ve tvaru $\bar{a}$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = a$,

3. pro φ ve tvaru $\mathbb{x} \approx \mathbb{y}$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \boldsymbol{v}(\mathbb{x}) \approx_y \boldsymbol{v}(\mathbb{y})$,
pro φ ve tvaru $\mathbb{x} \approx \mathfrak{c}$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \boldsymbol{v}(\mathbb{x}) \approx_y \mathfrak{c}^{\boldsymbol{\mathcal{D}}}$,
kde v obou případech požadujeme, aby $\lambda(y) = \lambda(rd(\mathbb{x}))$,

4. pro φ ve tvaru $\psi \otimes \chi$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \|\psi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} \otimes \|\chi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}}$,
pro φ ve tvaru $\psi \mathbin{\mathbb{\wedge}} \chi$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \|\psi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} \wedge \|\chi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}}$,
pro φ ve tvaru $\psi \mathbin{\mathbb{\vee}} \chi$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \|\psi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} \vee \|\chi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}}$,
pro φ ve tvaru $\psi \Rightarrow \chi$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \|\psi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} \to \|\chi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}}$,

5. pro φ ve tvaru $\Delta\psi$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \begin{cases} 1 & \text{pokud } \|\psi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = 1 \\ 0 & \text{jinak,} \end{cases}$

6. pro φ ve tvaru $(\forall \mathbb{x} \in \mathsf{R})\psi$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \bigwedge \|\psi\|^{rd'}_{\boldsymbol{\mathcal{D}},\boldsymbol{w}}$,
pro φ ve tvaru $(\exists \mathbb{x} \in \mathsf{R})\psi$ položíme $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \bigvee \|\psi\|^{rd'}_{\boldsymbol{\mathcal{D}},\boldsymbol{w}}$,
kde v obou případech uvažujeme ohodnocení $\boldsymbol{w}$, pro které máme $\boldsymbol{w} =_{\mathbb{x}} \boldsymbol{v}$ a $\boldsymbol{w}(\mathbb{x}) \in \mathsf{R}^{\boldsymbol{\mathcal{D}}}$. Dále uvažujeme deklaraci rozsahu $rd'\colon X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \boldsymbol{\Lambda})$ takovou, kde pro každé $\mathbb{y} \neq \mathbb{x}$ máme $rd'(\mathbb{y}) = rd(\mathbb{y})$ a $rd'(\mathbb{x}) = \mathsf{R}$.

Pravdivostní hodnota atomické formule $\mathfrak{r}(y_1{:}\,\mathbb{x}_1, \ldots, y_n{:}\,\mathbb{x}_n)$ z bodu 1 je tedy (při ohodnocení $\boldsymbol{v}$) definována jako stupeň, ve kterém příslušná n-tice patří do relace, jež slouží k interpretaci relačního symbolu $\mathfrak{r}$.

Podmínka $\lambda(y) = \lambda(rd(\mathbb{x}))$ v bodě 3 předchozí definice zajišťuje kompatibilitu použité relace podobnosti. Víme, že pro některé $y' \in Y$ platí $rd(\mathbb{x})^{\boldsymbol{\mathcal{D}}} \subseteq D_{y'}$, díky bezpečnosti formule φ vzhledem k rd platí $rd(\mathbb{y})^{\boldsymbol{\mathcal{D}}} \subseteq D_{y'}$ i $\mathfrak{c}^{\boldsymbol{\mathcal{D}}} \in D_{y'}$. Z podmínky $\lambda(y) = \lambda(rd(\mathbb{x})) = \lambda(y')$ plyne, že relace $\approx_y$ a $\approx_{y'}$ jsou totožné a tedy použitá relace $\approx_y$ je s hodnotami $\boldsymbol{v}(\mathbb{x}), \boldsymbol{v}(\mathbb{y}), \mathfrak{c}^{\boldsymbol{\mathcal{D}}}$ kompatibilní.

3.4.4. Dotazování v doménovém relačním kalkulu

Nyní na základě bezpečných formulí definujeme dotazování v relačním kalkulu. Uvažujme instanci databáze $\boldsymbol{\mathcal{D}}$ nad $\langle \mathbb{R}, \varrho \rangle$, jejíž domény respektují typovou deklaraci $\boldsymbol{\Lambda} = \langle \Lambda, \lambda \rangle$ pro Y a $\mathfrak{C}$. Dále uvažujme formuli φ, která je bezpečná vzhledem k $rd : X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \boldsymbol{\Lambda})$. Výraz $y{:}\mathbb{x}$ pro $y \in Y$, $\mathbb{x} \in X$ a $\lambda(y) = \lambda(rd(\mathbb{x}))$ nazveme *cílová komponenta*. Konečnou množinu vzájemně různých cílových komponent $\mathcal{T} = \{y_1{:}\mathbb{x}_1, \ldots, y_n{:}\mathbb{x}_n\}$ nazveme *cílová množina*. [3]

Pro $\mathcal{T} = \{y_1{:}\mathbb{x}_1, \ldots, y_n{:}\mathbb{x}_n\}$, $I = \{1, \ldots, n\}$ a φ definujeme relaci $\mathcal{T}\varphi^{rd,\boldsymbol{\mathcal{D}}}$ nad relačním schématem $R = \{y_1, \ldots, y_n\}$ takto: [3]

$$\left(\mathcal{T}\varphi^{rd,\boldsymbol{\mathcal{D}}}\right)(r) = \begin{cases} \bigvee\{\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} \mid \forall i \in I\colon \boldsymbol{v}(\mathbb{x}_i) = r(y_i)\}, & \text{pro } r \in \prod_{y_i \in R} rd(\mathbb{x}_i)^{\boldsymbol{\mathcal{D}}}, \\ 0 & \text{jinak,} \end{cases}$$

kterou nazveme *výsledek dotazu* $\mathcal{T}\varphi$ *v* $\boldsymbol{\mathcal{D}}$ *pod* rd. Formule φ může mít i jiné volné proměnné než ty, které se nacházejí v cílové množině, a pro různá ohodnocení těchto proměnných může být tato formule splněna v různých stupních. Na ohodnocení proměnných mimo cílovou množinu nám nezáleží, požadujeme pouze, aby při ohodnocení, jehož část přiřazující hodnoty proměnným z cílové množiny je pevně daná, byla formule pravdivá v co nejvyšším stupni.

Proto při výpočtu ranku pro danou n-tici r procházíme všechna $\boldsymbol{\mathcal{D}}$-ohodnocení $\boldsymbol{v}$ taková, že objektovým proměnným z cílové množiny tato ohodnocení $\boldsymbol{v}$ pevně přiřazují hodnoty příslušných atributů z n-tice r. Na ohodnocení ostatních proměnných neklademe žádné podmínky. Následně z množiny pravdivostních hodnot $\|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}}$ při uvažovaných ohodnoceních $\boldsymbol{v}$ vypočítáme supremum (na lineárně uspořádané množině pravdivostních hodnot maximum – lze říci, že maximalizujeme pravdivostní hodnotu formule φ přes všechna ohodnocení splňující výše uvedenou podmínku).

Nicméně pro každou bezpečnou formuli φ a cílovou množinu $\mathcal{T}$ existuje bezpečná formule ψ, jejíž volné proměnné jsou právě proměnné z cílové množiny $\mathcal{T}$ a pro každou instanci databáze $\boldsymbol{\mathcal{D}}$ platí $\mathcal{T}\psi^{rd,\boldsymbol{\mathcal{D}}} = \mathcal{T}\varphi^{rd,\boldsymbol{\mathcal{D}}}$ (výsledky jsou stejné) a navíc platí

$$\left(\mathcal{T}\psi^{rd,\boldsymbol{\mathcal{D}}}\right)(r) = \|\varphi\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}}$$

pro ohodnocení $\boldsymbol{v}$ taková, že pro každé r máme $\forall i \in I\colon \boldsymbol{v}(\mathbb{x}_i) = r(y_i)$. [3]

3.5. Vzájemný vztah základních dotazovacích jazyků

Ačkoliv má relační kalkul k dispozici plnou sílu predikátové logiky a relační algebra jen několik vybraných operací, oba dotazovací jazyky mají stejnou vyjadřovací sílu a jsou ekvivalentní. Ke každému dotazu v relačnímu kalkulu existuje dotaz v jazyce založeném na relační algebře takový, že výsledky obou dotazů jsou totožné, a naopak. Důkazy obou tvrzení jsou konstruktivní a poskytují algoritmus pro převod dotazů mezi oběma jazyky. Hlavní myšlenky obou důkazů jsou nastíněny níže.

První věta, umožňující přechod z relačního kalkulu do jazyka založeného na relační algebře, má následující znění. Mějme formuli φ, která je bezpečná vzhledem k $rd : X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \mathbf{\Lambda})$, a cílovou množinu $\mathcal{T}$. Potom existuje RA-výraz E_φ nad $\langle \mathbb{R}, \varrho \rangle$ takový, že $\mathcal{T}\psi^{rd,\boldsymbol{\mathcal{D}}} = E_\varphi^{\boldsymbol{\mathcal{D}}}$ pro libovolnou instanci databáze $\boldsymbol{\mathcal{D}}$ nad $\langle \mathbb{R}, \varrho \rangle$, jejíž domény respektují typovou deklaraci $\mathbf{\Lambda}$. [3]

Důkaz je veden strukturální indukcí přes podformule ψ formule φ. Prokáže se, že pro každou podformuli ψ máme:

- cílovou množinu $\mathcal{T}_\psi = \{x_1{:}\,\mathbb{x}_1, \ldots, x_n{:}\,\mathbb{x}_n\}$, kde názvy atributů x_i vycházejí z názvů proměnných $\mathbb{x}_i$, dále proměnné $\{\mathbb{x}_1, \ldots, \mathbb{x}_n\}$ jsou volné v ψ a pro každé $i = 1, \ldots, n$, pokud je $\mathbb{x}_i$ volná proměnná v φ, potom $\lambda(x_i) = \lambda(y)$ pro $y{:}\,\mathbb{x}_i \in \mathcal{T}$,
- deklaraci rozsahu $rd_\psi \colon X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \mathbf{\Lambda})$,
- RA-výraz F_ψ takový, že $\mathcal{T}_\psi\psi^{rd_\psi,\boldsymbol{\mathcal{D}}} = F_\psi^{\boldsymbol{\mathcal{D}}}$ pro instanci databáze $\boldsymbol{\mathcal{D}}$ nad $\langle \mathbb{R}, \varrho \rangle$ respektující $\mathbf{\Lambda}$. Tedy hodnota F_ψ v instanci databáze $\boldsymbol{\mathcal{D}}$ odpovídá výsledku dotazu $\mathcal{T}_\psi\psi$ v $\boldsymbol{\mathcal{D}}$.

Jelikož φ je sama sobě podformulí, i pro tuto formuli máme F_φ, pro který platí $\mathcal{T}_\varphi\varphi^{rd_\varphi,\boldsymbol{\mathcal{D}}} = F_\varphi^{\boldsymbol{\mathcal{D}}}$. Hledaný RA-výraz E_φ z F_φ získáme přejmenováním atributů. [3]

Každá proměnná $\mathbb{x}$, jež je volná v ψ, může být v φ buď volná, anebo vázaná ve výrazu ($\forall\mathbb{x}{\in}\mathsf{R}$) nebo ($\exists\mathbb{x}{\in}\mathsf{R}$). Pro ψ a proměnnou $\mathbb{x}$, jež je volná v ψ, můžeme zavést deklaraci rozsahu ve tvaru [3]:

$$\mathsf{R}_\mathbb{x} = \begin{cases} rd(\mathbb{x}) & \text{pokud je } \mathbb{x} \text{ volná ve } \varphi, \\ \mathsf{R} & \text{pokud je } \mathbb{x} \text{ vázaná ve } \varphi \text{ ve výrazech } (\forall\mathbb{x}{\in}\mathsf{R}) \text{ nebo } (\exists\mathbb{x}{\in}\mathsf{R}). \end{cases}$$

Jak bylo uvedeno dříve v 3.4.1. (str. 31), pro každý atribut $y \in Y$ stejného typu jako $\mathsf{R}_\mathbb{x}$ existuje RA-výraz $E_{\mathsf{R}_\mathbb{x},y}$, jehož vyhodnocením v instanci databáze $\boldsymbol{\mathcal{D}}$ získáme relaci $E^{\boldsymbol{\mathcal{D}}}_{\mathsf{R}_\mathbb{x},y}$ nad relačním schématem $\{y\}$. Tato relace je v jednoznačné korespondenci s $\mathsf{R}_\mathbb{x}^{\boldsymbol{\mathcal{D}}}$, tedy s množinou hodnot, které je podle deklarace rozsahu $\mathsf{R}_\mathbb{x}$ možné za proměnnou $\mathbb{x}$ dosazovat. Pro zjednodušení RA-výraz $E_{\mathsf{R}_\mathbb{x},y}$ označujeme jako $E_{\mathbb{x},y}$. [3]

Význam RA-výrazů $E_{\mathbb{x},y}$ je následující. Při vyhodnocování dotazů v relačním kalkulu máme pro objektové proměnné předepsané deklarace rozsahu, které omezují množinu hodnot, jež je možné za tyto proměnné dosadit. Potřebujeme zajistit stejný výsledek i pro RA-výraz, který z formule daného dotazu relačního kalkulu konstruujeme. Abychom tohoto dosáhli, pro volné proměnné vyskytující se v dané formuli vytvoříme RA-výraz $\bowtie_{i=1,\ldots,n} E_{\mathbb{x}_i,y_i}$. Nyní každá n-tice z kartézského součinu $\bowtie_{i=1,\ldots,n} E^{\boldsymbol{\mathcal{D}}}_{\mathbb{x}_i,y_i}$ odpovídá jednomu z přípustných ohodnocení proměnných $\mathbb{x}_1, \ldots, \mathbb{x}_n$.

Nyní je nutné prokázat existenci RA-výrazů pro jednotlivé tvary podformule ψ. Celý důkaz včetně zdůvodnění lze nalézt v [3]. Pro ilustraci jsou níže uvedeny vybrané RA-výrazy pro podformule ψ:

- Pro ψ ve tvaru $\mathbb{r}(y_1{:}\,\mathbb{x}_1, \ldots, y_n{:}\,\mathbb{x}_n)$ položíme

$$F_\psi = \rho_{h(y_i)=x_i}(\mathbb{r}) \bowtie E_{\mathbb{x}_1,x_1} \bowtie \cdots \bowtie E_{\mathbb{x}_n,x_n},$$

- pro ψ ve tvaru $\mathbb{x}_1 \approx \mathbb{x}_2$ položíme

$$F_\psi = \sigma_{x_1 \approx x_2}(E_{\mathbb{x}_1,x_1} \bowtie E_{\mathbb{x}_2,x_2}),$$

- pro ψ ve tvaru $(\chi \otimes \vartheta)$ označme $X_\chi = \{\mathbb{x}_1, \ldots, \mathbb{x}_n, \mathbb{y}_1, \ldots, \mathbb{y}_p\}$ a $X_\vartheta = \{\mathbb{x}_1, \ldots, \mathbb{x}_n, \mathbb{z}_1, \ldots, \mathbb{z}_q\}$ volné proměnné v χ a ϑ. Z indukčního předpokladu plyne existence RA-výrazů F_χ a F_ϑ pro formule χ a ϑ. Pro ψ položíme

$$F_\psi = (F_\chi \bowtie E_{\mathbb{z}_1,z_1} \bowtie \cdots \bowtie E_{\mathbb{z}_q,z_q}) \otimes (F_\theta \bowtie E_{\mathbb{y}_1,y_1} \bowtie \cdots \bowtie E_{\mathbb{y}_p,y_p}),$$

- pro ψ ve tvaru $(\forall \mathbb{x} \in \mathsf{R})\chi$ uvažujme množinu $\{\mathbb{x}, \mathbb{x}_1, \ldots, \mathbb{x}_n\}$ volných proměnných v χ. Z indukčního předpokladu plyne existence RA-výrazu F_χ pro formuli χ. Pro ψ položíme

$$F_\psi = F_\chi \div^G E_{\mathbb{x},x}, \text{kde } G = E_{\mathbb{x}_1,x_1} \bowtie \cdots \bowtie E_{\mathbb{x}_n,x_n}.$$

  Z definice dělení tedy pro n-tice r nad schématem $\{x_1, \ldots, x_n\}$ máme:

$$(F_\psi)(r) = \bigwedge_{s \in Tupl(\{x\})} ((E_{\mathbb{x}_1,x_1} \bowtie \cdots \bowtie E_{\mathbb{x}_n,x_n})(r) \otimes (E_{\mathbb{x},x}(s) \to F_\chi(rs)))$$

Druhá věta, pomocí které lze přejít z jazyka založeného na relační algebře do relačního kalkulu, je v následujícím tvaru. Mějme RA-výraz E nad $\langle \mathbb{R}, \varrho \rangle$. Potom existují deklarace rozsahu $rd_E \colon X \to \mathrm{Rd}(\mathbb{R}, \varrho, \mathfrak{C}, \boldsymbol{\Lambda})$, formule φ_E, která je bezpečná vzhledem k rd_E a cílová množina $\mathcal{T}_E$ tak, že platí $E^{\boldsymbol{\mathcal{D}}} = \mathcal{T}_E \varphi_E^{rd_E,\boldsymbol{\mathcal{D}}}$ pro libovolnou instanci databáze $\boldsymbol{\mathcal{D}}$, jejíž domény respektují $\boldsymbol{\Lambda}$ a všechny relace podobností splňují vlastnost separability. [3]

Důkaz je založen na strukturální indukci přes RA-výrazy. V důkazu se navíc používá následující úvaha. Pro RA-výraz E a příslušnou formuli φ_E s volnými proměnnými $\{\mathtt{y}_1, \ldots, \mathtt{y}_n\}$ definujeme formuli $\varphi^\star_E$ předpisem

$$(\exists \mathtt{y}'_1 \in rd_E(\mathtt{y}_1)) \cdots (\exists \mathtt{y}'_n \in rd_E(\mathtt{y}_n))(\Delta(\mathtt{y}_1 \approx \mathtt{y}'_1 \otimes \cdots \otimes \mathtt{y}_n \approx \mathtt{y}'_n) \otimes \varphi'_E),$$

kde φ'_E vznikne z φ_E nahrazením všech výskytů volných proměnných $\mathtt{y}_i$ proměnnými $\mathtt{y}'_i$. Formule $\varphi^\star_E$ má stejné volné proměnné jako φ_E a navíc díky separabilitě platí [3]:

$$\|\varphi^\star_E\|^{rd}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}} = \begin{cases} \|\varphi_E\|^{rd_E}_{\boldsymbol{\mathcal{D}},\boldsymbol{v}}, & \text{pokud } \boldsymbol{v}(\mathtt{y}_i) \in rd_E(\mathtt{y}_i)^{\boldsymbol{\mathcal{D}}} \text{ pro každé } i = 1, \ldots, n, \\ 0, & \text{jinak.} \end{cases}$$

Tímto postupem lze „zakódovat“ deklaraci rozsahu přímo do formule. Pro danou formuli poté můžeme uvažovat širší deklaraci rozsahu, aniž bychom rozšířili množinu ohodnocení proměnných, při kterých je tato formule pravdivá.

Nyní je nutné pro jednotlivé tvary RA-výrazů ukázat odpovídající dotazy v relačním kalkulu. Celý důkaz je opět možné nalézt v [3]. Opět pro ilustraci jsou níže uvedeny formule pro vybrané RA-výrazy, ve kterých je cílová množina a deklarace rozsahu je zapsána zkráceně jako $\{\ldots, y_i : \mathtt{y}_i \in \mathsf{R}, \ldots\}$:

- Pro E ve tvaru $\mathtt{r} \in \mathbb{R}$ uvažujeme tento dotaz

$$\{y_1 : \mathtt{y}_1 \in \{\mathtt{r}(y_1)\}, \ldots, y_n : \mathtt{y}_n \in \{\mathtt{r}(y_n)\}\}\mathtt{r}(y_1 : \mathtt{y}_1, \ldots, y_n : \mathtt{y}_n),$$

- pro E ve tvaru $E_1 \cup E_2$ máme z indukčního předpokladu existenci dotazů pro E_1 a E_2. Oba RA-výrazy jsou nad stejným relačním schématem a tedy φ_{E_1} i φ_{E_2} mají stejné volné proměnné. Deklaraci rozsahu pro φ_E definujeme jako sjednocení deklarací rozsahu φ_{E_1} a φ_{E_2}, což odpovídá tomu, že relace $(E_1 \cup E_2)^{\boldsymbol{\mathcal{D}}}$ obsahuje hodnoty z obou relací $E_1^{\boldsymbol{\mathcal{D}}}$ a $E_2^{\boldsymbol{\mathcal{D}}}$. Protože jsme použili obecnější deklaraci rozsahu, než mají φ_{E_1} a φ_{E_2}, musíme obě formule nahradit formulemi $\varphi^\star_{E_1}$ a $\varphi^\star_{E_2}$. Výsledný dotaz uvažujeme ve tvaru:

$$\{y_1 : \mathtt{y}_1 \in rd_{E_1}(\mathtt{y}_1) \cup rd_{E_2}(\mathtt{y}_1), \ldots, y_n : \mathtt{y}_n \in rd_{E_1}(\mathtt{y}_n) \cup rd_{E_2}(\mathtt{y}_n)\}\varphi^\star_{E_1} \mathbin{\mathbb{V}} \varphi^\star_{E_2}.$$

Z jistého úhlu pohledu je možné na operace relační algebry nahlížet jako na pečlivě vybrané formule predikátové logiky, které tvoří základní stavební kameny, pomocí kterých je možné sestavit libovolný dotaz, jehož výsledky je současně možné charakterizovat libovolnými formulemi relačního kalkulu. V obou případech tvoří základ dotazování predikátová logika, která je tedy pro relační model nepostradatelná.

4. Operace relačního dělení

Aby byla relační algebra schopna pokrýt veškeré dotazy, které je možné vyjádřit v relačním kalkulu, musí obsahovat dostatek vhodných operací. Pokud tomu tak je a oba jazyky mají stejnou vyjadřovací sílu, potom hovoříme o tzv. *relační úplnosti* relační algebry vzhledem k relačnímu kalkulu.

Klasický relační model je postaven na klasické dvouhodnotové logice, ve které je díky zákonu dvojí negace možné vzájemně definovat kvantifikátory. Všeobecný kvantifikátor je tak možné definovat pomocí existenčního jako $(\forall x)\varphi \equiv \neg(\exists x)\neg\varphi$. Pro úplnost klasické relační algebry tak stačí, aby tato algebra obsahovala operaci odpovídající jen jednomu kvantifikátoru. Operaci, která odpovídá druhému kvantifikátoru, je poté možné odvodit. Standardně se za základní operaci považuje projekce π, která odpovídá existenčnímu kvantifikátoru, a operace dělení je poté odvozena užitím projekce, spojení a rozdílu.

Zobecněný relační model je založen na logice, ve které obecně zákon dvojí negace neplatí a není tak možné kvantifikátory vzájemně definovat. Aby byla zobecněná relační algebra relačně úplná, musí kromě operace tvořící protějšek existenčního kvantifikátoru obsahovat i operaci odpovídající univerzálnímu kvantifikátoru – tedy relační dělení.

Operaci relačního dělení poprvé zmiňuje Codd již roku 1972 v jednom ze zakládajících článků relačního modelu [13]. Coddova definice relačního dělení trpěla několika problémy, které byly v průběhu času vyřešeny novými definicemi této operace [16]. Nicméně některé současné články [5] [6] [7] [22] [36], které se zabývají různými rozšířeními relačního modelu a uvažují i operaci relačního dělení, stále vycházejí z původní chybné definice a její problémy přejímají či v důsledku snahy se těmto problémům vyhnout trpí jinými [44].

4.1. Coddovo dělení

Operaci relačního dělení zavedl poprvé Codd ve svém článku z roku 1972 [13], který se zabýval relační úplností relační algebry. Codd tuto operaci charakterizoval jako algebraický protějšek ke všeobecnému kvantifikátoru a použil ji v důkazu relační úplnosti. Již v tomto článku však poznamenal, že operaci relačního dělení je možné odvodit.

V následujícím textu budeme pro přehlednost uvažovat výraz RS jako zkratku označující relační schéma $R \cup S$, pro které platí $R \cap S = \emptyset$. Nyní můžeme uvést definici Coddova relačního dělení. Mějme relace dělence $\mathcal{D}_1$ nad RS a dělitele $\mathcal{D}_2$ nad S, potom výsledek dělení je nad schématem R a operace je definována jako [16]

$$\mathcal{D}_1 \div_{\text{Codd}} \mathcal{D}_2 = \pi_R(\mathcal{D}_1) \setminus \pi_R((\pi_R(\mathcal{D}_1) \bowtie \mathcal{D}_2) \setminus \mathcal{D}_1).$$

Pro snazší představu uvažujme následující příklad. Předpokládejme, že atributy z R představují výrobce a atributy z S výrobky. Potom $\mathcal{D}_1$ můžeme

chápat jako relaci zachycující vztah „výrobce vyrábí produkt“. Relace $\mathcal{D}_2$ potom představuje produkty, které musí daný výrobce všechny vyrábět, aby mohl být zahrnut ve výsledku dělení. Podle definice pracuje operace takto:

1. Nejprve vypočítáme kartézský součin $\mathcal{D}_{all} = \pi_R(\mathcal{D}_1) \bowtie \mathcal{D}_2$, který obsahuje všechny možnosti párování „výrobce“ a „požadovaný produkt“.

2. Potom od kartézského součinu množinově odečteme relaci $\mathcal{D}_1$, čímž získáme relaci $\mathcal{D}_{\neg produce} = \mathcal{D}_{all} \setminus \mathcal{D}_1$, kterou můžeme interpretovat jako „výrobce nevyrábí produkt“. Výrobci, kteří vyrábějí všechny produkty z $\mathcal{D}_2$, se v této relaci nebudou vyskytovat.

3. Projekce $\pi_R(\mathcal{D}_{\neg produce})$ potom obsahuje výrobce, kteří nevyrábí některý z požadovaných produktů.

4. V posledním kroku množinovým odečtením výrobců, kteří nevyrábí některý z požadovaných produktů, od všech výrobců vyskytujících se v $\mathcal{D}_1$, získáme požadovanou relaci výrobců, kteří vyrábějí všechny produkty z $\mathcal{D}_2$.

Výsledek Coddova dělení lze množinově charakterizovat následovně (správnost této charakterizace plyne z obecnějšího případu, který je dokázán později v 4.5. (str. 42)):

$$\mathcal{D}_1 \div_{\text{Codd}} \mathcal{D}_2 = \{r \in \text{Tupl}(R) \mid r \in \pi_R(\mathcal{D}_1) \text{ a } \forall s \in \text{Tupl}(S)\colon (s \in \mathcal{D}_2 \Rightarrow rs \in \mathcal{D}_1)\},$$

kde relaci $\pi_R(\mathcal{D}_1)$ lze považovat za rozsah či universum pro toto dělení, jelikož vždy platí $\mathcal{D}_1 \div \mathcal{D}_2 \subseteq \pi_R(\mathcal{D}_1)$.

Jak bylo uvedeno dříve, s operací Coddova relačního dělení se pojí několik problémů. Méně závažný problém se týká omezení kladeného na relační schémata vstupních relací, kdy požadujeme, aby relační schéma dělitele $\mathcal{D}_2$ bylo podmnožinou relačního schématu dělence $\mathcal{D}_1$, ačkoliv k tomu není důvod. [16]

Závažnějším problémem je fakt, že existuje případ, kdy Coddovo dělení svým chováním neodpovídá všeobecnému kvantifikátoru. V krajním případě, kdy je relace dělitele $\mathcal{D}_2$ prázdná, je podmínka $\forall s \in \text{Tupl}(S)\colon (s \in \mathcal{D}_2 \Rightarrow rs \in \mathcal{D}_1)$ triviálně splněna pro každou n-tici $r \in \text{Tupl}(R)$. Uvážíme-li příklad s výrobci a produkty, můžeme říct, že každý výrobce triviálně vyrábí všechny produkty z prázdné množiny produktů, a to včetně výrobců, kteří nevyrábí žádný produkt. Takové výrobce, kteří nevyrábí žádný produkt, můžeme v databázi uvažovat (mohou být v relaci „výrobci“), ale nemohou být v relaci $\mathcal{D}_1$ reprezentující vztah „výrobce vyrábí produkt“. Výsledek Coddova dělení je však omezen pouze na n-tice z $\pi_R(\mathcal{D}_1)$, a tedy ačkoliv i pro výrobce nevyrábějící produkty je podmínka triviálně splněna, ve výsledku Coddova dělení nebudou zahrnuti. Ve výsledku Coddova dělení tedy mohou být pouze výrobci, kteří vyrábějí alespoň jeden produkt. Coddovo dělení tedy namísto dotazu „výrobci, kteří vyrábějí všechny produkty z $\mathcal{D}_2$“ odpovídá dotazu „výrobci, kteří vyrábějí alespoň jeden produkt a přitom všechny produkty z $\mathcal{D}_2$“. [16]

Přímé zobecnění Coddova relačního dělení lze uvažovat ve tvaru

$$(\mathcal{D}_1 \div_{\text{RCodd}} \mathcal{D}_2)(r) = \bigvee_{s \in \text{Tupl}(S)} (\mathcal{D}_1(rs)) \otimes \bigwedge_{s \in \text{Tupl}(S)} (\mathcal{D}_2(s) \to \mathcal{D}_1(rs))$$

pro každé $r \in \text{Tupl}(R)$. Toto zobecnění přejímá oba výše uvedené problémy.

4.2. Dělení podle [3]

V přístupu založeném na fuzzy logice v úzkém slova smyslu se uvažuje dělení, které umožňuje explicitně definovat rozsah dělení, a tak řeší závažnější problém Coddova dělení. Dělení je definováno takto. Uvažujme relace $\mathcal{D}_1$, $\mathcal{D}_2$ a $\mathcal{D}_3$ nad relačními schématy po řadě RS, S a R. Potom dělení $\mathcal{D}_1 \div^{\mathcal{D}_3} \mathcal{D}_2$ relace $\mathcal{D}_1$ relací $\mathcal{D}_2$ nad universem $\mathcal{D}_3$ je nad relačním schématem R a definujeme jej jako [3]

$$\begin{aligned}(\mathcal{D}_1 \div^{\mathcal{D}_3} \mathcal{D}_2)(r) &= \bigwedge_{s \in \text{Tupl}(S)} (\mathcal{D}_2(s) \twoheadrightarrow^{\mathcal{D}_3(r)} \mathcal{D}_1(rs)) \\ &= \bigwedge_{s \in \text{Tupl}(S)} (\mathcal{D}_3(r) \otimes (\mathcal{D}_2(s) \to \mathcal{D}_1(rs)))\end{aligned}$$

pro každé $r \in \text{Tupl}(R)$. Jistě platí $\mathcal{D}_1 \div^{\mathcal{D}_3} \mathcal{D}_2 \subseteq \mathcal{D}_3$. Druhý problém Coddova dělení je tak vyřešen. Pro příklad uvažujme, že relace $\mathcal{D}_3$ představuje výrobce. Pro prázdnou relaci dělitele máme $(\mathcal{D}_2(s) \to \mathcal{D}_1(rs)) = 1$ pro všechna $s \in \text{Tupl}(S)$ a tedy

$$(\mathcal{D}_1 \div^{\mathcal{D}_3} \mathcal{D}_2)(r) = \bigwedge_{s \in \text{Tupl}(S)} (\mathcal{D}_3(r) \otimes 1) = \mathcal{D}_3(r)$$

pro každé $r \in \text{Tupl}(R)$. Ve výsledku dělení tak budou skutečně zahrnuti všichni výrobci.

4.3. Malé dělení

Malé dělení je operace, kterou navrhnuli Date a Darwen v [19], aby odstranili závážnější problém Coddova dělení [16]. Uvažujme relace dělence $\mathcal{D}_1$ nad R, dělitele $\mathcal{D}_2$ nad S a prostředníka $\mathcal{D}_3$ nad RS. Výsledek malého dělení je potom nad R a je definován takto [16]

$$\mathcal{D}_1 \div_{\text{Sml}}^{\mathcal{D}_3} \mathcal{D}_2 = \mathcal{D}_1 \setminus \pi_R((\mathcal{D}_1 \bowtie \mathcal{D}_2) \setminus \mathcal{D}_3).$$

Množinově teoretické vyjádření malého dělení je ve tvaru:

$$\mathcal{D}_1 \div_{\text{Sml}} \mathcal{D}_2 = \{r \in \text{Tupl}(R) \mid r \in \mathcal{D}_1 \text{ a } \forall s \in \text{Tupl}(S) \colon (s \in \mathcal{D}_2 \Rightarrow rs \in \mathcal{D}_3)\},$$

a jeho přímé zobecnění pro každé $r \in \text{Tupl}(R)$:

$$(\mathcal{D}_1 \div_{\text{RSml}}^{\mathcal{D}_3} \mathcal{D}_2)(r) = \mathcal{D}_1(r) \otimes \bigwedge_{s \in \text{Tupl}(S)} (\mathcal{D}_2(s) \to \mathcal{D}_3(rs)).$$

Z praktických důvodů byla později zavedena ještě rozšířená verze malého dělení, která klade méně restriktivní nároky na relační schémata vstupních operací. Tato verze je definována pro relace dělence nad RT, dělitele nad SU a prostředníka RSV jako

$$\mathcal{D}_1 \div^{\mathcal{D}_3}_{\text{GSml}} \mathcal{D}_2 = \mathcal{D}_1 \bar{\ltimes} ((\pi_R(\mathcal{D}_1) \bowtie \pi_S(\mathcal{D}_2)) \bar{\ltimes} \mathcal{D}_3),$$

kde operace *semidiference* $\bar{\ltimes}$ je definována jako $\mathcal{D}_1 \bar{\ltimes} \mathcal{D}_2 = \mathcal{D}_1 \setminus (\mathcal{D}_1 \ltimes \mathcal{D}_2)$ [16].

4.4. Toddovo a velké dělení

Toddovo dělení v principu vychází z Coddova dělení a odstraňuje jeho méně závažný problém – umožňuje dělit relace nad libovolnými relačními schématy. Mějme relace dělence $\mathcal{D}_1$ nad RS a dělitele $\mathcal{D}_2$ nad ST, kde může platit i $S = \emptyset$. Výsledek Toddova dělení je potom nad relačním schématem RT a je ve tvaru [16]

$$\mathcal{D}_1 \div_{\text{Todd}} \mathcal{D}_2 = (\pi_R(\mathcal{D}_1) \bowtie \pi_T(\mathcal{D}_2)) \bar{\ltimes} ((\pi_R(\mathcal{D}_1) \bowtie \mathcal{D}_2) \bar{\ltimes} (\mathcal{D}_1 \bowtie \mathcal{D}_2)).$$

Množinově teoretické vyjádření je potom ve tvaru:

$$\mathcal{D}_1 \div_{\text{Todd}} \mathcal{D}_2 = \{rt \in \text{Tupl}(RT) \mid \underbrace{r \in \pi_R(\mathcal{D}_1) \text{ a } t \in \pi_T(\mathcal{D}_2)}_{rt \in ((\pi_R(\mathcal{D}_1)) \bowtie (\pi_T(\mathcal{D}_2))} \text{ a } \forall s \in \text{Tupl}(S) : (st \in \mathcal{D}_2 \Rightarrow rs \in \mathcal{D}_1)\},$$

a příslušné zobecnění pro každé $rt \in \text{Tupl}(RT)$:

$$(\mathcal{D}_1 \div_{\text{RTodd}} \mathcal{D}_2)(rt) = \underbrace{\bigvee_{s_1 \in \text{Tupl}(S)} (\mathcal{D}_1(rs_1)) \otimes \bigvee_{s_2 \in \text{Tupl}(S)} (\mathcal{D}_2(s_2 t))}_{((\pi_R(\mathcal{D}_1)) \bowtie (\pi_T(\mathcal{D}_2)))(rt)} \otimes \bigwedge_{s \in \text{Tupl}(S)} (\mathcal{D}_2(st) \to \mathcal{D}_1(rs)).$$

Toddovo dělení včetně jeho zobecnění však trpí podobnými problémy s prázdnými relacemi jako Coddovo dělení. Proto, ve stejném smyslu jako malé dělení, Date a Darwen navrhli velké dělení, které tento problém řeší. [16]

Velké dělení je definováno pro relace dělence $\mathcal{D}_1$ nad R, dělitele $\mathcal{D}_2$ nad T, prvního prostředníka $\mathcal{D}_3$ nad RS a druhého prostředníka $\mathcal{D}_4$ nad ST. Výsledek je nad relačním schématem RT a je dán předpisem [16]

$$\mathcal{D}_1 \div^{\mathcal{D}_3, \mathcal{D}_4}_{\text{Grt}} \mathcal{D}_2 = (\mathcal{D}_1 \bowtie \mathcal{D}_2) \bar{\ltimes} ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \bar{\ltimes} \mathcal{D}_3).$$

Výsledek velkého dělení lze množinově charakterizovat jako

$$\mathcal{D}_1 \div^{\mathcal{D}_3, \mathcal{D}_4}_{\text{Grt}} \mathcal{D}_2 = \{rt \in \text{Tupl}(RT) \mid \underbrace{r \in \mathcal{D}_1 \text{ a } t \in \mathcal{D}_2}_{rt \in (\mathcal{D}_1 \bowtie \mathcal{D}_2)} \text{ a } \forall s \in \text{Tupl}(S) : (st \in \mathcal{D}_4 \Rightarrow rs \in \mathcal{D}_3)\}.$$

Příslušné zobecnění pro každé $rt \in \mathrm{Tupl}(RT)$ je ve tvaru:

$$(\mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\mathrm{RGrt}} \mathcal{D}_2)(rt) = \underbrace{\mathcal{D}_1(r) \otimes \mathcal{D}_2(t)}_{(\mathcal{D}_1 \ltimes \mathcal{D}_2)(rt)} \otimes \bigwedge_{s \in \mathrm{Tupl}(S)} (\mathcal{D}_4(st) \to \mathcal{D}_3(rs)).$$

Velké dělení odstraňuje oba hlavní problémy Coddova relačního dělení. Pro praktické účely se opět uvažuje rozšířená verze velkého dělení s méně restriktivními nároky na relační schémata vstupních operací. Tato verze je definována pro relace dělence nad RU, dělitele nad TV a prostředníky RSW a STX. Výsledek dělení se poté uvažuje nad schématem $RTUV$ a je dán jako [3]

$$\mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\mathrm{GGrt}} \mathcal{D}_2 = (\mathcal{D}_1 \bowtie \mathcal{D}_2) \bar{\ltimes} ((\pi_R(\mathcal{D}_1) \bowtie \pi_{ST}(\mathcal{D}_4)) \bar{\ltimes} \mathcal{D}_3).$$

4.5. Darwenovo dělení

Nejobecnější definici relačního dělení navrhl Darwen. Darwenovo dělení koncepčně vychází z velkého dělení, ale na relační schémata neklade žádné požadavky. Mějme relace dělence $\mathcal{D}_1$ nad X_1, dělitele nad X_2 a prostředníky $\mathcal{D}_3$ a $\mathcal{D}_4$ nad X_3 a X_4. Výsledek Darwenova dělení je nad $X_1 \cup X_2$ (X_1 a X_2 nemusí být disjunktní) a je definován jako [16]

$$\mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\mathrm{Darw}} \mathcal{D}_2 = (\mathcal{D}_1 \bowtie \mathcal{D}_2) \ \bar{\ltimes} \ ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \ \bar{\ltimes} \ \mathcal{D}_3).$$

Ještě než odvodíme množinově teoretické vyjádření, prokážeme pomocné tvrzení charakterizující výsledek semidiference. Nechť jsou R, S, T relační schémata taková, že $R \cap S = R \cap T = S \cap T = \emptyset$. Uvažujme relace $\mathcal{D}_1$ nad RS a $\mathcal{D}_2$ nad ST. Pro $rs \in \mathrm{Tupl}(RS)$ platí následující tvrzení:

$$rs \in (\mathcal{D}_1 \ \bar{\ltimes} \ \mathcal{D}_2) \Leftrightarrow rs \in \mathcal{D}_1 \wedge \neg(\exists t \in \mathrm{Tupl}(T) \colon st \in \mathcal{D}_2).$$

Tvrzení plyne přímo z definice semidiference:

$$\begin{aligned}
rs \in \mathcal{D}_1 \ \bar{\ltimes} \ \mathcal{D}_2 &\Leftrightarrow rs \in (\mathcal{D}_1 \setminus \pi_{RS}(\mathcal{D}_1 \ltimes \mathcal{D}_2)) \\
&\Leftrightarrow rs \in \mathcal{D}_1 \wedge \neg(rs \in \pi_{RS}(\mathcal{D}_1 \ltimes \mathcal{D}_2)) \\
&\Leftrightarrow rs \in \mathcal{D}_1 \wedge \neg(rs \in (\mathcal{D}_1 \ltimes \pi_S(\mathcal{D}_2))) \\
&\Leftrightarrow rs \in \mathcal{D}_1 \wedge \neg(rs \in \mathcal{D}_1 \wedge s \in \pi_S(\mathcal{D}_2)) \\
&\Leftrightarrow rs \in \mathcal{D}_1 \wedge (\neg(rs \in \mathcal{D}_1) \vee \neg(s \in \pi_S(\mathcal{D}_2))) \\
&\Leftrightarrow \underbrace{(rs \in \mathcal{D}_1 \wedge \neg(rs \in \mathcal{D}_1))}_{\text{vždy nepravda}} \vee (rs \in \mathcal{D}_1 \wedge \neg(s \in \pi_S(\mathcal{D}_2))) \\
&\Leftrightarrow rs \in \mathcal{D}_1 \wedge \neg(s \in \pi_S(\mathcal{D}_2)) \\
&\Leftrightarrow rs \in \mathcal{D}_1 \wedge \neg(\exists t \in \mathrm{Tupl}(T) : st \in \mathcal{D}_2)
\end{aligned}$$

Toto tvrzení lze ekvivalentně vyjádřit jako:

$$rs \in (\mathcal{D}_1 \ \bar{\ltimes} \ \mathcal{D}_2) \Leftrightarrow rs \in \mathcal{D}_1 \wedge \neg(\exists s't \in \mathrm{Tupl}(ST) \colon \pi_S(rs) = \pi_S(s't) \wedge s't \in \mathcal{D}_2).$$

Dále uvažujme relace $\mathcal{D}_1$ a $\mathcal{D}_2$ nad obecnými relačními schématy R_1 a R_2. Řekneme, že n-tice $r_1 \in \mathcal{D}_1$ a $r_2 \in \mathcal{D}_2$ jsou *spojitelné* (ozn. $r_1 \between r_2$), pokud platí $\pi_{R_1 \cap R_2}(r_1) = \pi_{R_1 \cap R_2}(r_2)$.

Pro další úvahy poznamenejme, že pro obecná relační schémata R_1, R_2 stačí položit $R = R_1 \setminus R_2$, $S = R_1 \cap R_2$ a $T = R_2 \setminus R_1$. Takto definovaná relační schémata R, S, T jsou po dvou disjunktní a splňují předpoklad pomocného tvrzení. Pro relace $\mathcal{D}_1$ nad $R_1 = R \cup S$, $\mathcal{D}_2$ nad $R_2 = S \cup T$ a $r_1 \in \mathrm{Tupl}(R_1)$ tedy máme:

$$\begin{aligned} r_1 \in (\mathcal{D}_1 \mathbin{\bar{\ltimes}} \mathcal{D}_2) \\ &\Leftrightarrow r_1 \in \mathcal{D}_1 \wedge \neg(\exists r_2 \in \mathrm{Tupl}(R_2) \colon \pi_{R_1 \cap R_2}(r_1) = \pi_{R_1 \cap R_2}(r_2) \wedge r_2 \in \mathcal{D}_2) \\ &\Leftrightarrow r_1 \in \mathcal{D}_1 \wedge \neg(\exists r_2 \in \mathcal{D}_2 \colon r_1 \between r_2) \end{aligned}$$

Tedy n-tice r_1 patří do semidiference $(\mathcal{D}_1 \mathbin{\bar{\ltimes}} \mathcal{D}_2)$, právě když patří do $\mathcal{D}_1$ a současně neexistuje n-tice $r_2 \in \mathcal{D}_2$, se kterou by byla r_1 spojitelná.

Nyní vyjádříme Darwenovo dělení pomocí množinově-teoretické definice, kterou poté snadno zobecníme. Uvažujme relace dělence $\mathcal{D}_1$ nad X_1, dělitele nad X_2 a prostředníky $\mathcal{D}_3$ a $\mathcal{D}_4$ nad X_3 a X_4. Darwenovo dělení lze potom vyjádřit jako

$$\mathcal{D}_1 \div_{\mathrm{Darw}}^{\mathcal{D}_3, \mathcal{D}_4} \mathcal{D}_2 = \{r_1 r_2 \in \mathcal{U} \mid \forall r_4 \in \mathcal{D}_4 \colon (r_1 r_2 \between r_4 \Rightarrow \exists r_3 \in \mathcal{D}_3 \colon r_1 r_4 \between r_3)\},$$

kde $\mathcal{U} = \mathcal{D}_1 \bowtie \mathcal{D}_2$. Tento fakt nyní dokážeme.

Mějme n-tice $r_1 \in \mathrm{Tupl}(R_1), r_2 \in \mathrm{Tupl}(R_2)$, které jsou spojitelné $(r_1 \between r_2)$. S využitím pomocného tvrzení pro obecná relační schémata máme:

$$\begin{aligned} & r_1 r_2 \in \mathcal{D}_1 \div_{\mathrm{Darw}}^{\mathcal{D}_3, \mathcal{D}_4} \mathcal{D}_2 \\ \Leftrightarrow\ & r_1 r_2 \in ((\mathcal{D}_1 \bowtie \mathcal{D}_2) \mathbin{\bar{\ltimes}} ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \mathbin{\bar{\ltimes}} \mathcal{D}_3)) \\ \Leftrightarrow\ & r_1 r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_1' r_4 \in ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \mathbin{\bar{\ltimes}} \mathcal{D}_3) \colon r_1 r_2 \between r_1' r_4) \\ \Leftrightarrow\ & r_1 r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_1' r_4 \in \mathrm{Tupl}(R_1 \cup R_4) \colon \\ & \pi_{(R_1 \cup R_2) \cap (R_1 \cup R_4)}(r_1 r_2) = \pi_{(R_1 \cup R_2) \cap (R_1 \cup R_4)}(r_1' r_4) \wedge r_1' r_4 \in ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \mathbin{\bar{\ltimes}} \mathcal{D}_3)) \end{aligned}$$

Nyní upravíme podmínku $\pi_{(R_1 \cup R_2) \cap (R_1 \cup R_4)}(r_1 r_2) = \pi_{(R_1 \cup R_2) \cap (R_1 \cup R_4)}(r_1' r_4)$:

$$\begin{aligned} & \pi_{(R_1 \cup R_2) \cap (R_1 \cup R_4)}(r_1 r_2) = \pi_{(R_1 \cup R_2) \cap (R_1 \cup R_4)}(r_1' r_4) \\ \Leftrightarrow\ & \pi_{R_1 \cup (R_2 \cap R_4)}(r_1 r_2) = \pi_{R_1 \cup (R_2 \cap R_4)}(r_1' r_4) \\ \Leftrightarrow\ & \pi_{R_1}(r_1 r_2) = \pi_{R_1}(r_1' r_4) \wedge \pi_{R_2 \cap R_4}(r_1 r_2) = \pi_{R_2 \cap R_4}(r_1' r_4) \\ \Leftrightarrow\ & r_1 = r_1' \wedge \pi_{R_2 \cap R_4}(\pi_{R_2}(r_1 r_2)) = \pi_{R_2 \cap R_4}(\pi_{R_4}(r_1' r_4)) \\ \Leftrightarrow\ & r_1 = r_1' \wedge \pi_{R_2 \cap R_4}(r_2) = \pi_{R_2 \cap R_4}(r_4) \\ \Leftrightarrow\ & r_1 = r_1' \wedge r_2 \between r_4 \end{aligned}$$

Výraz $\exists r_1' r_4 \in \mathrm{Tupl}(R_1 \cup R_4)$ lze navíc chápat jako

$$\exists r_1' \in \mathrm{Tupl}(R_1), \exists r_4 \in \mathrm{Tupl}(R_4) \colon r_1' \between r_4$$

Máme tedy:

$$
\begin{aligned}
& r_1r_2 \in \mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\text{Darw}} \mathcal{D}_2 \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r'_1 r_4 \in \text{Tupl}(R_1 \cup R_4): \\
& \pi_{(R_1\cup R_2)\cap(R_1\cup R_4)}(r_1r_2) = \pi_{(R_1\cup R_2)\cap(R_1\cup R_4)}(r'_1r_4) \wedge r'_1r_4 \in ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \bar{\ltimes} \mathcal{D}_3)) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r'_1 \in \text{Tupl}(R_1), \exists r_4 \in \text{Tupl}(R_4): \\
& (r'_1 \between r_4) \wedge (r_1 = r'_1) \wedge (r_2 \between r_4) \wedge (r'_1r_4 \in ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \bar{\ltimes} \mathcal{D}_3)))
\end{aligned}
$$

Díky podmínce $r_1 = r'_1$ není potřeba uvažovat $\exists r'_1 \in \text{Tupl}(R_1)$. Dále zjednodušíme podmínku $(r'_1 \between r_4) \wedge (r_2 \between r_4)$. Využijeme předpokladu, že n-tice r_1 a r_2 jsou spojitelné. Potom jistě platí $r_1 = \pi_{R_1}(r_1r_2)$, obdobně pro r_2. Z faktů $r_1 = r'_1$, spojitelnosti n-tic r_1 a r_2 a z definice spojitelnosti máme:

$$
\begin{aligned}
& r'_1 \between r_4 \wedge r_2 \between r_4 \\
\Leftrightarrow\ & r_1 \between r_4 \wedge r_2 \between r_4 \\
\Leftrightarrow\ & \pi_{(R_1\cap R_4)}(r_1) = \pi_{(R_1\cap R_4)}(r_4) \wedge \pi_{R_2\cap R_4}(r_2) = \pi_{R_2\cap R_4}(r_4) \\
\Leftrightarrow\ & \pi_{(R_1\cap R_4)}(\pi_{R_1}(r_1r_2)) = \pi_{(R_1\cap R_4)}(r_4) \wedge \pi_{R_2\cap R_4}(\pi_{R_2}(r_1r_2)) = \pi_{R_2\cap R_4}(r_4) \\
\Leftrightarrow\ & \pi_{(R_1\cap R_4)}(r_1r_2) = \pi_{(R_1\cap R_4)}(r_4) \wedge \pi_{R_2\cap R_4}(r_1r_2) = \pi_{R_2\cap R_4}(r_4) \\
\Leftrightarrow\ & \pi_{(R_1\cap R_4)\cup(R_2\cap R_4)}(r_1r_2) = \pi_{(R_1\cap R_4)\cup(R_2\cap R_4)}(r_4) \\
\Leftrightarrow\ & \pi_{(R_1\cup R_2)\cap R_4}(r_1r_2) = \pi_{(R_1\cup R_2)\cap R_4}(r_4) \\
\Leftrightarrow\ & r_1r_2 \between r_4
\end{aligned}
$$

Toto pozorování nyní aplikujeme a dále rozepíšeme zbývající semidiferenci. Potom postupnou úpravou získáme požadované tvrzení:

$$
\begin{aligned}
& r_1r_2 \in \mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\text{Darw}} \mathcal{D}_2 \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r'_1 \in \text{Tupl}(R_1), \exists r_4 \in \text{Tupl}(R_4): \\
& (r'_1 \between r_4) \wedge (r_1 = r'_1) \wedge (r_2 \between r_4) \wedge (r'_1r_4 \in ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \bar{\ltimes} \mathcal{D}_3))) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_4 \in \text{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_1r_4 \in ((\mathcal{D}_1 \bowtie \mathcal{D}_4) \bar{\ltimes} \mathcal{D}_3))) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_4 \in \text{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_1r_4 \in (\mathcal{D}_1 \bowtie \mathcal{D}_4)) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3)) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_4 \in \text{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_1 \in \mathcal{D}_1) \wedge (r_4 \in \mathcal{D}_4) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3)) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg((r_1 \in \mathcal{D}_1) \wedge (\exists r_4 \in \text{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_4 \in \mathcal{D}_4) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3))) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge (\neg(r_1 \in \mathcal{D}_1) \vee \neg(\exists r_4 \in \text{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_4 \in \mathcal{D}_4) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3))) \\
\Leftrightarrow\ & (r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(r_1 \in \mathcal{D}_1)) \vee (r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_4 \in \text{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_4 \in \mathcal{D}_4) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3)))
\end{aligned}
$$

Z předchozí strany máme:

$$
\begin{aligned}
& r_1r_2 \in \mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\text{Darw}} \mathcal{D}_2 \\
\Leftrightarrow\ & (r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(r_1 \in \mathcal{D}_1)) \vee (r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_4 \in \mathrm{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_4 \in \mathcal{D}_4) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3))) \\
\Leftrightarrow\ & ((r_1 \in \mathcal{D}_1) \wedge (r_1 \notin \mathcal{D}_1) \wedge (r_2 \in \mathcal{D}_2)) \vee (r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_4 \in \mathrm{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_4 \in \mathcal{D}_4) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3))) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_4 \in \mathrm{Tupl}(R_4): \\
& (r_1r_2 \between r_4) \wedge (r_4 \in \mathcal{D}_4) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3)) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge \neg(\exists r_4 \in \mathcal{D}_4 \colon (r_1r_2 \between r_4) \wedge \neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3)) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge (\forall r_4 \in \mathcal{D}_4 \colon \neg(r_1r_2 \between r_4) \vee \neg\neg(\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3)) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge (\forall r_4 \in \mathcal{D}_4 \colon \neg(r_1r_2 \between r_4) \vee (\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3)) \\
\Leftrightarrow\ & r_1r_2 \in (\mathcal{D}_1 \bowtie \mathcal{D}_2) \wedge (\forall r_4 \in \mathcal{D}_4 \colon r_1r_2 \between r_4 \Rightarrow (\exists r_3 \in \mathcal{D}_3 \colon r_1r_4 \between r_3)).
\end{aligned}
$$

Darwenovo dělení lze charakterizovat slovně takto. Daná n-tice r_1r_2 patří do výsledku dělení, právě když:

- r_1r_2 náleží do spojení $\mathcal{D}_1 \bowtie \mathcal{D}_2$,
- pro každou n-tici $r_4 \in \mathcal{D}_4$, která je spojitelná s r_1r_2, existuje n-tice $r_3 \in \mathcal{D}_3$, která je spojitelná s r_1r_4.

Nyní již snadno Darwenovo dělení zobecníme. Uvažujme relace dělence $\mathcal{D}_1$ nad X_1, dělitele nad X_2 a prostředníky $\mathcal{D}_3$ a $\mathcal{D}_4$ nad X_3 a X_4. Pro každou n-tici $r_1r_2 \in \mathrm{Tupl}(R_1 \cup R_2)$ máme

$$
\mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\text{RDarw}} \mathcal{D}_2(r_1r_2) = \mathcal{D}_1(r_1) \otimes \mathcal{D}_2(r_2) \otimes \left(\bigwedge_{\substack{r_4 \in \mathrm{Tupl}(R_4) \\ r_1r_2 \between r_4}} \mathcal{D}_4(r_4) \to \bigvee_{\substack{r_3 \in \mathrm{Tupl}(R_3) \\ r_1r_4 \between r_3}} \mathcal{D}_3(r_3) \right).
$$

4.6. Vlastnosti a vzájemný vztah operací relačního dělení

Pokud jako základní operaci uvažujeme libovolné relační dělení, které postačuje k prokázání relační úplnosti, potom všechna ostatní dělení jsou odvoditelná. Nicméně mezi některými operacemi existují zajímavé vztahy, které jsou uvedeny níže. Některé operace relační algebry je navíc možné vyjádřit jako speciální případ dělení.

4.6.1. Operace „být podmnožinou“

Operace „být podmnožinou“ je definována pro dvě relace $\mathcal{D}_1$ a $\mathcal{D}_2$ nad stejným relačním schématem R. Výsledkem této operace je relace $a_\emptyset$ nad prázdným

relačním schématem, kde $a \in L$ je stupeň, ve kterém platí $\mathcal{D}_1 \subseteq \mathcal{D}_2$. Tuto operaci je možné definovat pomocí dělení podle [3], malého, velkého i Darwenova dělení. Pro ilustraci uveďme definici například pomocí malého dělení:

$$(\mathcal{D}_1 \subseteq \mathcal{D}_2)(\emptyset) = 1_\emptyset \div^{\mathcal{D}_2}_{RSml} \mathcal{D}_1 = \overbrace{1_\emptyset(\emptyset)}^{=1} \otimes \bigwedge_{r \in \mathrm{Tupl}(R)} (\mathcal{D}_1(r) \to \mathcal{D}_2(r))$$
$$= \bigwedge_{r \in \mathrm{Tupl}(R)} (\mathcal{D}_1(r) \to \mathcal{D}_2(r))$$

4.6.2. Operace residua podle [3]

Tato operace je definována pro relace $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ nad stejným relačním schématem R. Tuto operaci lze chápat jako speciální případ Darwenova dělení:

$$\begin{aligned}(\mathcal{D}_1 \twoheadrightarrow^{\mathcal{D}_3} \mathcal{D}_2)(r) &= (\mathcal{D}_3 \div^{\mathcal{D}_2,\mathcal{D}_1}_{\mathrm{RDarw}} 1_\emptyset)(r) \\ &= \mathcal{D}_3(r) \otimes 1_\emptyset(\emptyset) \otimes \left(\bigwedge_{\substack{r' \in \mathrm{Tupl}(R) \\ r \emptyset \not\between r'}} \mathcal{D}_1(r') \to \bigvee_{\substack{r'' \in \mathrm{Tupl}(R) \\ r r' \not\between r''}} \mathcal{D}_2(r'') \right) \\ &= \mathcal{D}_3(r) \otimes (\mathcal{D}_1(r) \to \mathcal{D}_2(r)) \\ &= \mathcal{D}_1(r) \twoheadrightarrow^{\mathcal{D}_3(r)} \mathcal{D}_2(r)\end{aligned}$$

Ve výše uvedeném vztahu jsou n-tice r, r', r'' nad stejným relačním schématem R a tedy, aby mohly být spojitelné, musejí si být rovny. Díky tomu jsou infimum a supremum počítány vždy jen z jednoho prvku a nemusíme je uvažovat.

Pomocí Darwenova dělení je možné definovat obecnější operaci ve stylu residua, kdy uvolníme požadavky na relační schémata vstupních relací a budeme požadovat pouze $R_1 \subseteq R_3$ a $R_2 \subseteq R_3$. V důsledku tak lze definovat operace $\otimes$-průniku (položíme $\mathcal{D}_1 = 1_\emptyset$) či a-negaci (položíme $\mathcal{D}_2 = a_\emptyset$).

4.6.3. Vztah dělení podle [3] a malého dělení

Definice obou dělení jsou téměř identické, liší se pouze v pozici relace, která určuje universum dělení. V případě dělení podle [3] se tato relace nachází uvnitř infima, u malého dělení pak vně.

Pokud uvažujeme strukturu pravdivostních hodnot, která splňuje podmínku prelinearity, tedy pro každé $a, b \in L$ platí $(a \to b) \vee (b \to a) = 1$, nebo divisibility, tedy pro každé $a, b \in L$ platí $a \wedge b = a \otimes (a \to b)$, potom jsou obě operace ekvivalentní.

Z prelinearity nebo divisibility plyne distributivita součinu vůči infimu, tedy pro všechna $a, b, c \in L$ platí $a \otimes (b \wedge c) = (a \otimes b) \wedge (a \otimes c)$ [2]. Navíc díky konečnosti

relací počítáme infimum pouze z konečně mnoha prvků různých od 1. Využitím těchto faktů pro $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ nad RS, S a R máme [44]

$$\begin{aligned}(\mathcal{D}_1 \div^{\mathcal{D}_3} \mathcal{D}_2)(r) &= \bigwedge_{s \in \mathrm{Tupl}(S)} (\mathcal{D}_3(r) \otimes (\mathcal{D}_2(s) \to \mathcal{D}_1(rs))) \\ &= \mathcal{D}_3(r) \otimes \bigwedge_{s \in \mathrm{Tupl}(S)} (\mathcal{D}_2(s) \to \mathcal{D}_1(rs)) \\ &= (\mathcal{D}_3 \div^{\mathcal{D}_1}_{\mathrm{RSml}} \mathcal{D}_2)(r)\end{aligned}$$

Pokud pro $r \in \mathrm{Tupl}(R)$ platí $\mathcal{D}_3(r) \in \{0, 1\}$, potom jsou obě operace dělení ekvivalentní bez ohledu na uvažovanou strukturu pravdivostních hodnot. [44]

4.6.4. Vztah malého a velkého dělení

Malé dělení lze vyjádřit jako speciální případ velkého dělení. Pro relace $\mathcal{D}_1$ nad R, $\mathcal{D}_2$ nad S a $\mathcal{D}_3$ nad RS máme:

$$(\mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_2}_{\mathrm{RGrt}} 1_\emptyset)(r) = \mathcal{D}_1(r) \otimes 1_\emptyset \otimes \bigwedge_{s \in \mathrm{Tupl}(S)} (\mathcal{D}_2(s) \to \mathcal{D}_3(rs)) = (\mathcal{D}_1 \div^{\mathcal{D}_3}_{\mathrm{RSml}} \mathcal{D}_2)(r).$$

4.6.5. Vztah velkého a Darwenova dělení

Velké dělení je možné přímočaře vyjádřit jako speciální případ Darwenova dělení. Uvažujme relace $\mathcal{D}_1$ nad R, $\mathcal{D}_2$ nad T, $\mathcal{D}_3$ nad RS a $\mathcal{D}_4$ nad ST, potom máme:

$$\begin{aligned}\mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\mathrm{RDarw}} \mathcal{D}_2(rt) &= \mathcal{D}_1(r) \otimes \mathcal{D}_2(t) \otimes \left(\bigwedge_{\substack{st' \in \mathrm{Tupl}(ST) \\ rt \between st'}} \mathcal{D}_4(st') \to \bigvee_{\substack{r's' \in \mathrm{Tupl}(RS) \\ rst' \between r's'}} \mathcal{D}_3(r's') \right) \\ &= \mathcal{D}_1(r) \otimes \mathcal{D}_2(t) \otimes \left(\bigwedge_{s \in \mathrm{Tupl}(S)} \mathcal{D}_4(st) \to \mathcal{D}_3(rs) \right) \\ &= \mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\mathrm{RGrt}} \mathcal{D}_2(rt)\end{aligned}$$

Podobně jako v 4.6.2. si je třeba uvědomit, že části n-tic t, t' v operaci infima jsou nad stejným relačním schématem a aby mohly být spojitelné, musí si být rovny. Navíc $R \cap S = \emptyset$, proto podmínka $rt \between st'$ je ekvivalentní podmínce $t = t'$. Z toho důvodu stačí namísto $st' \in \mathrm{Tupl}(ST)$ uvažovat pouze $s \in \mathrm{Tupl}(S)$. Stejný argument použijeme pro supremum.

4.6.6. Vyjádření operace dělení pomocí operace „být podmnožinou“

Pokud budeme považovat operaci „být podmnožinou“ za základní, pak lze dělení vyjádřit pomocí spojení, selekce na rovnost a operace „být podmnožinou“.

Pro úplnost uvedeme definici operace „být podmnožinou“. Uvažujme relace $\mathcal{D}_1$ a $\mathcal{D}_2$ nad stejným relačním schématem R. Výsledkem operace $\mathcal{D}_1 \subseteq \mathcal{D}_2$ je vždy některá relace $a_\emptyset$. Operaci definujeme takto:

$$(\mathcal{D}_1 \subseteq \mathcal{D}_2)(\emptyset) = \bigwedge_{r \in \mathrm{Tupl}(R)} (\mathcal{D}_1(r) \to \mathcal{D}_2(r)).$$

Operaci selekce na rovnost $\sigma_{y=d}(\mathcal{D})$ je možné vyjádřit pomocí spojení s relací singletonu jako $\mathcal{D} \bowtie [y\!:\!d]$ [3].

Nyní pro relace $\mathcal{D}_1$ nad $R = \{y_1, \ldots, y_n\}$, $\mathcal{D}_2$ nad S a $\mathcal{D}_3$ nad RS uvažujme:

$$\begin{aligned}
& (\mathcal{D}_1 \div^{\mathcal{D}_3}_{\mathrm{RSml}} \mathcal{D}_2)(r) \\
= \;& \mathcal{D}_1(r) \otimes \bigwedge_{s \in \mathrm{Tupl}(S)} (\mathcal{D}_2(s) \to \mathcal{D}_3(rs)) \\
= \;& \mathcal{D}_1(r) \otimes \bigwedge_{s \in \mathrm{Tupl}(S)} (\mathcal{D}_2(s) \to (\pi_S(\mathcal{D}_3 \bowtie [y_1\!:\!r(y_1)] \bowtie \cdots \bowtie [y_n\!:\!r(y_n)]))(s)) \\
= \;& \mathcal{D}_1(r) \otimes (\mathcal{D}_2 \subseteq (\pi_S(\mathcal{D}_3 \bowtie [y_1\!:\!r(y_1)] \bowtie \cdots \bowtie [y_n\!:\!r(y_n)])))(\emptyset) \\
= \;& (\mathcal{D}_1 \bowtie (\mathcal{D}_2 \subseteq (\pi_S(\mathcal{D}_3 \bowtie [y_1\!:\!r(y_1)] \bowtie \cdots \bowtie [y_n\!:\!r(y_n)]))))(r)
\end{aligned}$$

Označme relaci $\mathcal{D}_3 \bowtie [y_1\!:\!r(y_1)] \bowtie \cdots \bowtie [y_n\!:\!r(y_n)]$ jako $\mathcal{D}^r_3$. Poznamenejme, že platí $\pi_S(\mathcal{D}^r_3)(s) = \mathcal{D}^r_3(rs)$, protože po selekci na rovnost zůstane v relaci $\mathcal{D}^r_3$ ke každému s jediné r. A tedy máme $\pi_S(\mathcal{D}^r_3)(s) = \bigvee_{r \in \mathrm{Tupl}(R)} \mathcal{D}^r_3(rs) = \mathcal{D}^r_3(rs)$. Projekce ve výše uvedeném vztahu hraje technickou roli, kdy pouze upravuje relační schéma $\mathcal{D}^r_3$ tak, aby bylo možné aplikovat operaci „být podmnožinou.“ Relaci $\mathcal{D}^r_3$ lze chápat jako obraz relace (*image relation*) $\mathcal{D}_3$ pro r tak, jak jej definuje Date v [16].

4.7. Podpora dělení v dotazovacích jazycích pro uživatele

Nejznámější dotazovací jazyk `SQL` operaci relačního dělení nepodporuje v žádné podobě [16]. Dotazy s všeobecným charakterem je nutné vyjádřit pomocí vnořených dotazů. Pro ilustraci opět uvažujme příklad s výrobci a produkty, kdy tabulka VP představuje „výrobce vyrábí produkt“ a tabulka P potom požadované výrobky. Coddovo dělení je možné pomocí vnořených dotazů vyjádřit takto:

```
select distinct VP1.vyrobce from VP as VP1 where not exists
    (select * from P where not exists
        (select * from VP as VP2 where
            (VP1.vyrobce = VP2.vyrobce) and (VP2.produkt = P.produkt)));
```

V literatuře [10] lze nalézt přístup založený na agregaci, který však není přesným vyjádřením Coddova dělení, jelikož při prázdné relaci dělitele jako

výsledek vrací prázdnou relaci. Dotaz je ve tvaru [10]:

> **select** VP.vyrobce **from** VP, P **where** VP.produkt = P.produkt
> **group by** VP.vyrobce
> **having count** (VP.produkt) = (**select count**(produkt) **from** P);

Jazyk `Tutorial D` podporuje malé a velké dělení pomocí výrazů v následujícím tvaru [16]:

- malé dělení: $\mathcal{D}_1 \div^{\mathcal{D}_3}_{\text{Sml}} \mathcal{D}_2 \equiv \mathcal{D}_1$ DIVIDEBY $\mathcal{D}_2$ PER $(\mathcal{D}_3)$
- velké dělení: $\mathcal{D}_1 \div^{\mathcal{D}_3,\mathcal{D}_4}_{\text{Grt}} \mathcal{D}_2 \equiv \mathcal{D}_1$ DIVIDEBY $\mathcal{D}_2$ PER $(\mathcal{D}_3, \mathcal{D}_4)$

Podle Date [16] je celá oblast nepřehledná a pro uživatele dotazovacího jazyka je náročné znát sémantiku jednotlivých operací relačního dělení. Proto Date navrhuje upustit od podpory DIVIDEBY v `Tutorial D`. Namísto dělení navrhuje zavést podporu pro porovnávání relací a *image relations*, pomocí kterých je možné formulovat dotazy se všeobecným charakterem intuitivněji.

V kontextu, kdy je možné uvažovat jednotlivé n-tice (např. v „těle“ výrazu WHERE), Date pojem obrazu relace $\mathcal{D}$ definuje takto [16]. Mějme aktuálně uvažovanou n-tici $r \in \text{Tupl}(R)$, kde $R = \{y_1, \ldots, y_n\}$ a relaci $\mathcal{D}$ nad S, jejíž obraz počítáme. Obrazem potom rozumíme relaci

$$!!\mathcal{D} = \pi_{S \setminus R}(\mathcal{D} \bowtie [y_1 : r(y_1)] \bowtie \cdots \bowtie [y_n : r(y_n)]).$$

S využitím *image relations* a porovnávání relací je možné například malé dělení vyjádřit takto [16]:

$$\mathcal{D}_1 \text{ DIVIDEBY } \mathcal{D}_2 \text{ PER } (\mathcal{D}_3) \equiv \mathcal{D}_1 \text{ WHERE } \mathcal{D}_2 \subseteq !!\mathcal{D}_3$$

Při výpočtu $\mathcal{D}_1$ WHERE $\mathcal{D}_2 \subseteq !!\mathcal{D}_3$ se postupně zpracovávají n-tice z $\mathcal{D}_1$, pro každou n-tici se vypočítá obraz $\mathcal{D}_3$ a pokud platí $\mathcal{D}_2 \subseteq !!\mathcal{D}_3$, potom je aktuálně zpracovávaná n-tice zahrnuta do výsledku dotazu.

Uvažujme opět příklad s výrobci ($\mathcal{D}_1$) a požadovanými produkty ($\mathcal{D}_2$), ve kterém relace $\mathcal{D}_3$ reprezentuje vztah „výrobce vyrábí produkt“. Pro daného výrobce z $\mathcal{D}_1$ lze $!!\mathcal{D}_3$ chápat jako produkty, které tento výrobce vyrábí. Výsledek dotazu $\mathcal{D}_1$ WHERE $\mathcal{D}_2 \subseteq !!\mathcal{D}_3$ je možné intuitivně charakterizovat takto. Do výsledku budou zahrnuti takoví výrobci, pro které platí, že požadované produkty tvoří podmnožinu jejich portfolia, tedy jinými slovy, jsou to výrobci, kteří vyrábí všechny požadované výrobky.

Při návrhu praktického jazyka s podporou ranků je možné podporu dotazů s všeobecným charakterem vyřešit právě pomocí *image relations*. Díky tomu, že je operaci dělení možné odvodit pomocí operace „být podmnožinou“, bude takto navržený jazyk relačně úplný.

Díky rankům se může operace relačního dělení stát z uživatelského pohledu zajímavější, než je v klasickém případě. Opět uvažujme příklad s výrobci a produkty. Ranky můžeme u jednotlivých relací interpretovat takto:

- relace výrobců $\mathcal{D}_1$ – rank $\mathcal{D}_1(r)$ můžeme interpretovat jako stupeň, ve kterém preferujeme daného výrobce,
- relace požadovaných produktů $\mathcal{D}_2$ – rank $\mathcal{D}_2(s)$ můžeme chápat jako stupeň charakterizující míru, ve které je daný produkt požadovaný (rank 1 = zcela, rank 0 = vůbec),
- relace „výrobce vyrábí produkt“ $\mathcal{D}_3$ – rank $\mathcal{D}_3(rs) = 1$ interpretujeme standardně jako „výrobce r vyrábí produkt s“, nižší rank je poté možno chápat jako „výrobce vyrábí jiný produkt, který lze považovat za náhradu produktu s, jejíž vhodnost je dána příslušným rankem“.

Výsledkem dělení potom budou výrobci, kteří jsou schopni dodat všechny produkty, které potřebujeme (případně dodají vhodný náhradní produkt), a navíc budou seřazeni podle jejich celkové vhodnosti.

5. Operace GROUP a UNGROUP

V relačním modelu (v moderním pojetí [17], [18], [20]) mohou domény obsahovat libovolné hodnoty. Můžeme uvažovat doménu všech celých čísel, řetězců, ale i zvukových nahrávek, fotografií, fuzzy množin a samozřejmě taky relací nad libovolným relačním schématem.

Ústředním pojmem relačního modelu jsou relace, proto je přirozené uvažovat operace, které umožňují mapovat relaci či její část na relaci ve smyslu hodnoty a naopak. Pro tento účel Date uvažuje operace `GROUP` a `UNGROUP`. Díky těmto operacím je možné s relacemi pracovat jako s daty. Naopak pro manipulaci s relacemi jako hodnotami není nutné zavádět žádné nové operace a lze využít standardních operací relační algebry (po provedení `UNGROUP`). Atributy, jejichž doménu tvoří právě relace, nazýváme *relačně-hodnotové atributy* (relation-valued attributes, RVAs). [18]

Relačně-hodnotové atributy je možné využít k seskupení souvisejících částí ve výsledku dotazu a tak dosáhnout přehlednější reprezentace výsledku dotazu. Pomocí relačně-hodnotových atributů je navíc možné logicky čistým způsobem nahradit vnější spojení, která využívají nerelační pojem `NULL`. Na druhou stranu, Date nedoporučuje používat relačně-hodnotové atributy v bázových relacích, jelikož jejich použití ve většině případů vede k problémům, kterými trpěly starší hierarchické modely. [18]

V zobecněném modelu mají relačně-hodnotové atributy a příslušné operace `GROUP` a `UNGROUP` ještě jeden důležitý význam. Díky nim je možné pracovat s ranky dané relace jako s daty. Relaci s ranky je možné převést na klasickou relaci, jejíž relační schéma je rozšířeno o relačně-hodnotový atribut rank. Pro každou n-tici r z relace s ranky potom v klasické relaci máme n-tici r', která vyjma původních hodnot z r obsahuje také relaci nad prázdným relačním schématem $a_\emptyset$, která odpovídá ranku n-tice r ve výchozí relaci s ranky.

5.1. První normální forma

Výše uvedené moderní pojetí domén je v rozporu s tradičním chápáním první normální formy, kdy se kromě jiného požaduje následující [18]:

> Hodnoty v doménách, nad kterými jsou následně definovány relace, musejí být *atomické* vzhledem k databázovému systému. Atomickými daty rozumíme taková data, která není možné dále pomocí databázového systému rozložit na menší části.

V tradičním chápání relační model požaduje, aby každá relace vyhovovala podmínkám první normální formy. Podle výše uvedené definice tak není možné uvažovat relace jako hodnoty. Vyvstává však otázka, jaká data je možné považovat za atomická – řetězce je možné pomocí funkce `SUBSTRING` rozložit na jednotlivé znaky, celá čísla je možné rozložit na součin prvočinitelů, anebo

přímo na jednotlivé bity. Navíc pokud je do databázového systému doprogramována funkce, která umožňuje rozložit doposud atomická data, tato data přestávají okamžikem přidání funkce být atomická. Date proto koncept atomicity a příslušnou definici první normální formy zavrhuje. První normální formu Date definuje takto: [18]

> Tabulka je v první normální formě, právě když je přímou a přesnou reprezentací nějaké relace.

Přesněji řečeno, příslušná tabulka musí splňovat následující podmínky:

1. nezáleží na uspořádání jednotlivých řádků
2. nezáleží na uspořádání jednotlivých sloupců
3. v tabulce se nevyskytují žádné duplicitní řádky
4. v každé „buňce“ tabulky (v průniku řádku a sloupce) je vždy právě jedna hodnota z příslušné domény
5. žádný řádek neobsahuje skrytou informaci, která by byla přístupná jen pomocí speciálních funkcí

V tomto moderním pojetí první normální formy nemá smysl hovořit o tom, zdali relace splňuje první normální formu, jelikož podmínky kladené první normální formou relace ze své matematické podstaty splňuje vždy. Na první normální formu je vhodné nahlížet jako na prostředek, jenž zajišťuje, aby tabulky, pomocí kterých obvykle reprezentujeme relace, opravdu byly přesnou reprezentací příslušných relací. V tomto pojetí první normální formy navíc nic nebrání použití libovolných domén tak, jak bylo popsáno v úvodu. [18]

V literatuře je možné narazit na přístupy (např. [14] [38] [40] [43]), které vycházejí z tradičního chápání první normální formy a z jejich pohledu rozšiřují relační model o možnost použití relačně-hodnotových atributů. Tyto přístupy se často označují jako „nested model“, „¬1NF“ či „NF^2“ (non-first normal form). Tyto přístupy je skutečně možné považovat za rozšíření relačního modelu, jelikož upravují relační operace tak, aby byly schopné pracovat i nad relacemi vyskytujícími se „uvnitř“ jiných relací. [18]

5.2. Zobecněné operace GROUP a UNGROUP

Pro definici těchto operací budeme potřebovat následující pojmy. Atribut, jehož doménu tvoří relace nad relačním schématem R, nazveme *atribut relačního typu* R. Nyní uvažujme n-tici t nad T s atributem $y \in T$ relačního typu R. Hodnotou atributu y v n-tici t (ozn. $t(y)$) je relace s ranky nad relačním schématem R. Pro n-tici r nad R umožníme zjistit stupeň její příslušnosti do relace $t(y)$. Tento

stupeň označíme $r \in t(y)$. Pokud je t nad jednoprvkovým relačním schématem, pak pro přehlednost namísto $r \in t(y)$ píšeme $r \in t$.

Nyní již můžeme přistoupit k definicím zobecněných variant příslušných operací. Nejprve uvedeme návrh definice zobecněné operace `GROUP`. Uvažujme relaci $\mathcal{D}$ nad relačním schématem R, dále relační schéma $S \subseteq R$ a atribut $y \notin R$ relačního typu S. Výsledek operace `GROUP` $\gamma_{y \leftarrow S}(\mathcal{D})$ je nad relačním schématem $(R \setminus S) \cup \{y\}$ a definujeme jej jako

$$(\gamma_{y \leftarrow S}(\mathcal{D}))(aX) = \nabla \left(\bigvee_{b \in \mathrm{Tupl}(S)} \mathcal{D}(ab) \right) \otimes \Delta \left(\bigwedge_{b \in \mathrm{Tupl}(S)} (\mathcal{D}(ab) \leftrightarrow b \in X) \right)$$

pro každou n-tici $a \in \mathrm{Tupl}(R \setminus S)$ a n-tici $X \in \mathrm{Tupl}(\{y\})$ (n-tice X je označena velkým písmenem pro zdůraznění faktu, že hodnotou jejího jediného atributu je relace). Výsledkem operace $\gamma_{y \leftarrow S}(\mathcal{D})$ je vždy klasická relace. Vlivem ∇ a Δ je levá i pravá strana součinu rovna 0 nebo 1 a pro každé $a \in L$ máme $0 \otimes a = a \otimes 0 = 0$ a $1 \otimes 1 = 1$. Proto pro každou n-tici $t \in (R \setminus S) \cup \{y\}$ platí $\gamma_{y \leftarrow S}(\mathcal{D})(t) \in \{0, 1\}$.

Levou stranu součinu $\nabla \left(\bigvee_{b \in \mathrm{Tupl}(S)} \mathcal{D}(ab) \right)$ si lze pro dané $a \in \mathrm{Tupl}(R \setminus S)$ představit takto

$$\nabla(\pi_{R \setminus S}(\mathcal{D}))(a) = \begin{cases} 1, & \text{v } \mathcal{D} \text{ existuje n-tice, jejíž část je tvořena n-ticí } a, \\ 0, & \text{jinak.} \end{cases}$$

Tento výraz slouží k ošetření speciálního případu, kdy n-tice a není částí žádné n-tice z $\mathcal{D}$. V tomto případě je pravá část součinu rovna 1 pro n-tici $X \in \mathrm{Tupl}(\{y\})$ takovou, že $X(y)$ je prázdná relace. Pokud bychom levou část součinu neuvažovali, pak bychom měli $\gamma_{y \leftarrow S}(\mathcal{D})(aX) = 1$ pro každé $a \in \mathrm{Tupl}(R \setminus S)$, které není částí žádné n-tice z $\mathcal{D}$ a X takové, že $X(y)$ je prázdná relace.

Pravá strana součinu je rovna 1 jen tehdy, když pro dané $a \in \mathrm{Tupl}(R \setminus S)$ a $X \in \mathrm{Tupl}(\{y\})$ platí, že stupeň, ve kterém je ab v $\mathcal{D}$, je roven stupni, ve kterém je b v $X(y)$, pro každé $b \in \mathrm{Tupl}(S)$. Tedy, $X(y)$ se musí rovnat relaci, která vznikne z $\mathcal{D}$ tak, že nejprve provedeme restrikci na rovnost s a, a z výsledku restrikce poté projekci na atributy S. Jinými slovy, $X(y)$ se musí rovnat obrazu relace (*image relation*) $\mathcal{D}$ pro n-tici a.

Nyní uvedeme návrh definice zobecněné operace `UNGROUP`. Uvažujme relaci $\mathcal{D}$ nad relačním schématem R a atribut $y \in R$ relačního typu S, pro které platí $(R \setminus \{y\}) \cap S = \emptyset$. Výsledek operace `UNGROUP` $\curlywedge_y(\mathcal{D})$ je potom nad $(R \setminus \{y\}) \cup S$ a definujeme jej jako

$$(\curlywedge_y(\mathcal{D}))(ab) = \bigvee_{X \in \mathrm{Tupl}(\{y\})} (\mathcal{D}(aX) \otimes b \in X)$$

pro každé $a \in \mathrm{Tupl}(R)$ a $b \in \mathrm{Tupl}(S)$.

Pro $\mathcal{D}$ nad relačním schématem R, schéma $S \subseteq R$ a atribut $y \notin R$ relačního typu S stejně jako v klasické relační algebře platí

$$\curlywedge_y(\gamma_{y \leftarrow S}(\mathcal{D})))(ab) = \mathcal{D}(ab)$$

pro každé $a \in \mathrm{Tupl}(R \setminus S)$ a $b \in \mathrm{Tupl}(S)$. Důkaz tvrzení provedeme rozborem případů. Pro libovolně zvolené $a \in \mathrm{Tupl}(R \setminus S)$ uvažujme následující případy:

1. Pro všechna $b \in \mathrm{Tupl}(S)$ máme $\mathcal{D}(ab) = 0$.

   Potom máme $(\gamma_{y \leftarrow S}(\mathcal{D}))(aX) = 0$ pro libovolné $X \in \mathrm{Tupl}(\{y\})$, protože pro levou stranu součinu v $\gamma_{y \leftarrow S}(\mathcal{D})$ platí $\nabla\left(\bigvee_{b \in \mathrm{Tupl}(S)} \mathcal{D}(ab)\right) = 0$. Potom ale $\curlywedge_y(\gamma_{y \leftarrow S}(\mathcal{D})))(ab) = 0 = \mathcal{D}(ab)$, protože $\bigvee_{X \in \mathrm{Tupl}(\{y\})}(\gamma_{y \leftarrow S}(\mathcal{D}))(aX) = 0$.

2. Pro některé $b \in \mathrm{Tupl}(S)$ máme $\mathcal{D}(ab) > 0$.

   Potom máme $(\gamma_{y \leftarrow S}(\mathcal{D}))(aX) = 1$ pro jediné $X \in \mathrm{Tupl}(\{y\})$, které je dáno takto. Pro každé $b \in \mathrm{Tupl}(S)$ máme $b \in X = \mathcal{D}(ab)$. Pro ostatní $X' \in \mathrm{Tupl}(\{y\})$ platí $(\gamma_{y \leftarrow S}(\mathcal{D}))(aX') = 0$. Potom

$$\begin{aligned}
\curlywedge_y(\gamma_{y \leftarrow S}(\mathcal{D})))(ab) &= \bigvee_{X'' \in \mathrm{Tupl}(\{y\})} (\mathcal{D}(aX'') \otimes b \in X'') \\
&= \mathcal{D}(aX) \otimes b \in X \\
&= 1 \otimes \mathcal{D}(ab) \\
&= \mathcal{D}(ab).
\end{aligned}$$

V tabulce 4. je možné nalézt jednoduché příklady použití operace `GROUP`. Část 4.a zobrazuje výchozí relaci s ranky $\mathcal{D}$. Obě zbylé části obsahují klasické relace. Relace $\gamma_{y\_rel \leftarrow \{y\}}(\mathcal{D})$ v 4.b představuje „běžné“ použití operace `GROUP`. Oproti tomu relace $\gamma_{rank \leftarrow \emptyset}(\mathcal{D})$ v části 4.c demonstruje fakt, že je díky `GROUP` možné s ranky pracovat jako s daty. Navíc každá relace $a_\emptyset$ je v jednoznačné korespondenci s příslušným stupněm $a \in L$. Díky těmto faktům je možné převést relace s ranky na klasické relace, vykonat nad nimi požadované dotazy v klasickém modelu a opět je pomocí `UNGROUP` převést zpět.

Pro demonstraci uvažujme jednoduchý příklad výpočtu spojení dvou relací. Mějme relace $\mathcal{D}_1$ nad RS a $\mathcal{D}_2$ nad ST. Označme si `R1` relaci, která vznikne z relace $\gamma_{rank1 \leftarrow \emptyset}(\mathcal{D}_1)$ nahrazením relací $a_\emptyset$ za příslušné hodnoty a, obdobně pro `R2`. Nyní můžeme použít např. `Rel` (za předpokladu, že jsme jej vybavili operátorem `OTIMES`, který realizuje výpočet $\otimes$):

WITH (Temp1 := R1 **JOIN** R2,
Temp2 := **EXTEND** Temp1 : {rank := rank1 **OTIMES** rank2}) :
Temp2 { **ALL BUT** rank1, rank2 }

Nyní již stačí ve výsledku klasického dotazu ve sloupci rank nahradit $a \in L$ zpět za $a_\emptyset$ a provést $\curlywedge_{\mathrm{rank}}(\mathcal{D})$. Pokud měla některá n-tice v atributu rank hodnotu $0_\emptyset$, pak po provedení `UNGROUP` tato n-tice „zanikne“ a ve výsledku nebude zahrnuta.

	x	y
1.0	1	1
0.8	1	2
0.4	2	2
1.0	3	2
0.3	3	4
0.1	3	5

(a) Výchozí relace $\mathcal{D}$

x	y_rel	
1		y
	1.0	1
	0.8	2
2		y
	0.4	2
3		y
	1.0	2
	0.3	4
	0.1	5

(b) Relace $\gamma_{y\_rel \leftarrow \{y\}}(\mathcal{D})$

$rank$	x	y
$(1.0)_\emptyset$	1	1
$(0.8)_\emptyset$	1	2
$(0.4)_\emptyset$	2	2
$(1.0)_\emptyset$	3	2
$(0.3)_\emptyset$	3	4
$(0.1)_\emptyset$	3	5

(c) Relace $\gamma_{rank \leftarrow \emptyset}(\mathcal{D})$

Tabulka 4.: Příklady použití operace GROUP

Závěr

Práce přinesla přehled přístupů, které se zabývají logickými modely dat s podporou podobnostních dotazů. Podrobně popsala přístup založený na fuzzy logice v úzkém slova smyslu, kdy se uvažuje zobecněný relační model, který je namísto klasické dvouhodnotové logiky založen na logice obecnější.

Jelikož ve fuzzy logice obecně není možné vzájemně definovat kvantifikátory, musí být v zobecněné relační algebře zahrnuta i operace relačního dělení. Práce se proto zabývala různými definicemi relačního dělení včetně jejich zobecněných variant. Dále ukázala, že je dělení možné vyjádřit pomocí operace „být podmnožinou“. Díky tomu je možné v praktickém jazyce s podporou podobnostních dotazů namísto dělení uživateli nabídnout bez ztráty relační úplnosti tzv. *image relations*, které jsou pro uživatele přirozenější a jejich použití intuitivnější. Práce dále popsala možnou interpretaci zobecněného relačního dělení, díky které by mohla být zobecněná varianta pro uživatele zajímavější než klasické relační dělení.

Dále práce nastínila možnou definici zobecněných variant operací `GROUP` a `UNGROUP`. Tyto operace umožňují převádět relace s ranky na klasické relace a zpět. Pomocí tohoto postupu je možné zobecněný relační model implementovat v klasickém databázovém systému.

Jelikož operace `GROUP` a `UNGROUP` používají pojmy, které na straně relačního kalkulu vyžadují predikátovou logiku druhého řádu, další výzkum se zaměří na prozkoumání důsledků tohoto faktu a na rozšíření důkazu relační úplnosti. V klasickém relačním modelu je navíc operace `GROUP` v úzkém vztahu s `EXTEND` a *image relations*, výzkum bude pokračovat i tímto směrem. Cílem dalšího výzkumu je dále efektivní implementace zobecněného relačního modelu.

Conclusions

This work described main approaches to the logical data models with support for similarity-based querying. The approach based on fuzzy logic in narrow sense was described in greater detail. In this approach the relational data model is based on more general logic than the classical two-valued one.

Since in general the quantifiers in fuzzy logic are not mutually definable, the relational division has to be considered as a fundamental operation in the generalized relational algebra. The work presented various definitions of this operation including their generalized forms. It also showed, that in a practical query language one can replace the division operation with image relations that offer more natural querying without losing relational completeness. The interpretation of ranks with respect to relational division was also considered and it is belived that the generalized division operation might be more appealing to users than the classical version of this operation.

This work also proposed definitions of generalized versions of `GROUP` and `UNGROUP` operations. These operations can be used to transform the generalized relation with ranks to the classical one and back. This procedure makes implementation of the generalized model in classical database system possible.

Since the `GROUP` and `UNGROUP` operations use notions that require second-order predicate logic on the side of relational calculus, further research should focus on the consequences of this fact and should extend the proof of relational completeness. In classical relational model there is a tight connection between `GROUP` and `EXTEND` with image relations. The research will continue in this direction as well. Effective implementation of generalized relational model will also be considered.

Reference

[1] Bao, Jun-Peng. Shen, Jun-Yi. Liu, Xiao-Domg. Liu, Hai-Yan. *Quick asymmetric text similarity measures.* Machine Learning and Cybernetics, 2003 International Conference on. Vol. 1. IEEE, 2003.

[2] Belohlavek, Radim. *Fuzzy relational systems: foundations and principles.* Kluwer Academic Publishers, 2002.

[3] Belohlavek, Radim. Vychodil, Vilem. *Relational Similarity-Based Databases Part 1: Foundations and Query Systems* (článek v přípravě, 79 str.)

[4] Bosc, Patrick. Pivert, Olivier. Smits, Grégory. *A model based on outranking for database preference queries.* Information Processing and Management of Uncertainty in Knowledge-Based Systems. Applications. Springer Berlin Heidelberg, 2010. 95-104.

[5] Bosc, Patrick. Dubois, Didier. Pivert, Olivier. Prade, Henri. *Flexible queries in relational databases—the example of the division operator.* Theoretical Computer Science, 171(1) (1997): 281-302.

[6] Bosc, Patrick. Pivert, Olivier. Rocacher, Daniel. *Characterizing the result of the division of fuzzy relations.* International journal of approximate reasoning 45.3 (2007): 511-530.

[7] Bosc, Patrick. *On the primitivity of the division of fuzzy relations.* Soft Computing 2.2 (1998): 35-47.

[8] Boutilier, Craig. Brafman, Ronen. Domshlak, Carmel. *CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements.* J. Artif. Intell. Res.(JAIR) 21 (2004): 135-191.

[9] Bruno, Nicolas. Gravano, Luis. Marian, Amelie. *Evaluating top-k queries over webaccessible databases.* In Proceedings of the 18th International Conference on Data Engineering, pages 369–380, 2002.

[10] Celko, Joe. *SQL for Smarties: Advanced SQL Programming.* Morgan Kaufmann, 2005.

[11] Codd, Edgar F. *A relational model of data for large shared data banks.* Communications of the ACM 13.6 (1970): 377-387.

[12] Codd, Edgar F. *A data base sublanguage founded on the relational calculus.* Proceedings of the 1971 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control. ACM, 1971.

[13] Codd, Edgar F. *Relational completeness of data base sublanguages.* Database Systems, Prentice-Hall, 1972.

[14] Colby, Latha S. *A recursive algebra for nested relations.* Information Systems 15.5 (1990): 567-582.

[15] Dalvi, Nilesh. Suciu, Dan. *Management of probabilistic data: foundations and challenges.* Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. ACM, 2007.

[16] Date, Chris J. Darwen, Hugh. *Database Explorations: Essays on The Third Manifesto and related topics.* Trafford Publishing, 2010.

[17] Date, Chris J. *Database in depth: relational theory for practitioners.* O'Reilly Media, Inc., 2005.

[18] Date, Chris J. *Date on database: writings 2000-2006.* Apress, 2006.

[19] Date, Chris J. Darwen, Hugh. *Into the great divide.*, Relational database writings 1991 (1989).

[20] Date, Chris J. *SQL and relational theory: how to write accurate SQL code.* O'Reilly Media, Inc., 2011.

[21] Date, Chris J. *The Database Relational Model: A Retrospective Review and Analysis.* Addison-Wesley, 2001.

[22] Dubois, Didier. Prade, Henri. *Semantics of quotient operators in fuzzy relational databases.* Fuzzy sets and systems 78.1 (1996): 89-93.

[23] Eduardo, Juan. Goncalves, Marlene. Tineo, Leonid. *A fuzzy querying system based on SQLf2 and SQLf3.* The XXX Latin-American Conference on Informatics. 2004.

[24] Garcia-Molina, Hector. Ullman, Jeffrey D. Widom, Jennifer. *Database Systems: The Complete Book (2 ed.).* Prentice Hall Press, Upper Saddle River, NJ, USA, 2008.

[25] Gottwald, Siegfried. *Mathematical fuzzy logics.* Bulletin of Symbolic Logic 14.02 (2008): 210-239.

[26] Chang, Kevin Chen-Chuan. Hwang, Seung-won. *Minimal probing: supporting expensive predicates for top-k queries.* Proceedings of the 2002 ACM SIGMOD international conference on Management of data. ACM, 2002.

[27] Li, Chengkai. *Enabling data retrieval: By ranking and beyond.* ProQuest, 2007.

[28] Koch, Christoph. *MayBMS: A system for managing large uncertain and probabilistic databases.* Managing and Mining Uncertain Data (2009): 149.

[29] Lacroix, Michel. Pirotte, Alain. *Domain-oriented relational languages.* Proceedings of the third international conference on Very large data bases-Volume 3. VLDB Endowment, 1977.

[30] Ilyas, Ihab F. Aref, Walid G. Elmagarmid, Ahmed K. *Supporting top-k join queries in relational databases.* The VLDB Journal—The International Journal on Very Large Data Bases 13.3 (2004): 207-221.

[31] Ilyas, Ihab F. Shah, Rahul. Aref, Walid G. Vitter, Jeffrey S. Elmagarmid, Ahmed. *Rank-aware query optimization.* Proceedings of the 2004 ACM SIGMOD international conference on Management of data. ACM, 2004.

[32] Maier, David. *The theory of relational databases.* Vol. 11. Rockville: Computer science press, 1983.

[33] Medina, J. M. Vila, M. A. Cubero, Juan-Carlos. Pons, O. *Towards the implementation of a generalized fuzzy relational database model.* Fuzzy Sets and Systems 75.3 (1995): 273-289.

[34] Orłowska, Ewa. Radzikowska, Anna M. *Double residuated lattices and their applications.* Relational Methods in Computer Science. Springer Berlin Heidelberg, 2002. 171-189.

[35] Ortony, Andrew. Vondruska, Richard J. Foss, Mark A. Jones, Lawrence E. *Salience, similes, and the asymmetry of similarity.* Journal of memory and language 24.5 (1985): 569-594.

[36] Pivert, Olivier. Bosc, Patrick. Fuzzy Preference Queries to Relational Databases. London: Imperial College Press, 2012.

[37] Rabitti, Fausto. Savino, Pasquale. *Retrieval of multimedia documents by imprecise query specification.* Advances in Database Technology—EDBT'90. Springer Berlin Heidelberg, 1990. 203-218.

[38] Roth, Mark A. Korth, Herry F. Silberschatz, Abraham. *Extended algebra and calculus for nested relational databases.* ACM Transactions on Database Systems (TODS) 13.4 (1988): 389-417.

[39] Sumathi, Sai. Esakkirajan S. *Fundamentals of relational database management systems.* Vol. 47. Springer, 2007.

[40] Thomas, Stan J., and Patrick C. Fischer. *Nested relational structures.* Advances in Computing Research 3 (1986): 269-307.

[41] Tolias, Yannis A. Panas, Stavros M. Tsoukalas, Lefteri H. *Generalized fuzzy indices for similarity matching.* Fuzzy Sets and Systems 120.2 (2001): 255-270.

[42] Tversky, Amos. *Features of Similarity* Psychological review 84.4 (1977).

[43] Van Gucht, Dirk. Fischer, Patrick C. *Multilevel nested relational structures.* Journal of Computer and System Sciences 36.1 (1988): 77-105.

[44] Vaverka, Ondrej. Vychodil, Vilem. *Relational Division in Ranked-Aware Databases: An Overview, Issues, and New Directions.* (článek přijat na konferenci MDAI 2014, 12 str.)