Kvalita relačního schématu, normalizace

Dva přístupy k návrhu struktury relačního schématu:

- normalizační teorie
  Metoda návrhu pomocí funkčních závislostí
- z konceptuálního schématu
  Metoda návrhu pomocí transformačních pravidel

Návrh relací - Kvalita schématu

Uvažujme relaci:
PROGRAM(NÁZEV_K, JMÉNO_F, ADRESA, DATUM)

- změní-li se adresa kina, je nutné ji měnit víckrát,
- nehraje-li kino zrovna nic, ztrácíme jeho adresu,
- chceme-li přidat nové kino s adresou, lze to jen když se tam hraje nějaký film.

Codd: Aktualizační anomálie

Jak to zlepšíme?

Normalizace dekompozicí

KINO(NÁZEV_K, ADRESA)
MÁ_NA_PROGRAMU(NÁZEV_K, JMÉNO_F, DATUM)
IO: MÁ_NA_PROGRAMU[NÁZEV_K] ⊆ KINO[NÁZEV_K]

Návrh relací - Kvalita schématu

- Hodnoty některých atributů **funkčně závisí** na hodnotách jiných atributů.
  - ke každému kinu existuje nejvýše jedna adresa
  - pro každé kino a film existuje nejvýše jedno datum, kdy dané kino má daný film na programu

NÁZEV_K → ADRESA,

{NÁZEV_K,JMÉNO_F} → DATUM,

Funkční závislosti

Návrh relací - Kvalita schématu

- vztah typ entity - typ entity ... relationship
- vztah typ entity - datový typ ... Atribut Ai
- vztah dom(Ai) - dom(Aj) ... **funkční závislost**

{R(A),F}

  - F ... jedna z možností zápisu IO
  - IO ... tvrzení o tom, které entice z D1x... xDn jsou přípustné

Integritní omezení (funkční závislosti) definují množinu přípustných relací R*, které mohou vzniknout podle schématu R(A)

Návrh relací - Kvalita schématu

Příklad:

Rozvrh (Přednáška,Učitel,Místnost,Hodina,Student, Známka)

Vnitropodnikové pravidlo:

Každá přednáška je přednášena právě/nejvýše jedním učitelem

Tvrzení k DB schématu:

K jedné hodnotě z dom(Přednáška) se přiřadí právě /nejvýše jedna hodnota z dom(Učitel)

Předmět $\rightarrow$ Učitel

zkraťme to: P $\rightarrow$ U

Návrh relací - Kvalita schématu

Nechť ve schématu ROZVRH jsou zakódovány aktualizační anomálie. Nahraďme schéma množinou schémat tak, aby výsledek měl “rozumné vlastnosti”

výchozí schéma: **R** = R (P,U,M,H,S,Z) ~ PUMHSZ

$\mathbf{R_I}$ = {PU, HMP, HUM, PSZ, HSM}

$\mathbf{R_{II}}$ = {PU, HSP, PSZ, HSM}

$\mathbf{R_{III}}$ = {PU, HSM, PSZ, HMP}

$\mathbf{R_{IV}}$ = {PU, HMP, PSZ, HSP}

$\mathbf{R_V}$ = {HMPU, PSZ, HSM}

$\mathbf{R_{VI}}$ = {PU, HMP, PSZ} $\mathbf{R_{VII}}$ = {PSUHM, PSZ}

Která množina schémat je lepší?

Návrh relací - Kvalita schématu

Odhalení FZ mezi atributy schématu

P	U	M	H	S	Z
Programování	Kryl	S7	Po9	Novák	2
Programování	Kryl	S3	Út3	Novák	2
Programování	Kryl	S7	Po9	Volák	3
Programování	Kryl	S3	Út3	Volák	3
Systémy	Král	S4	Po7	Zíka	1
Systémy	Král	S4	Po7	Tupý	2
Systémy	Král	S4	Po7	Novák	2
Systémy	Král	S4	Po7	Bílý	1

možná U $\not\rightarrow$ HM určitě

možná platí P $\rightarrow$ U,HM$\rightarrow$P,HU$\rightarrow$M, HS$\rightarrow$ M

a co toto: F S $\rightarrow$ Z

Návrh relací - *Funkční závislost*

Mějme schéma R(A), uvažujme $X \subseteq A$

X-hodnota:

Jsou-li atributy v **X** $\{X_1{:}dom(X_1),\ldots,X_n{:}dom(X_n)\}$,

pak **X-hodnotou** je libovolný prvek z kartézského součinu $dom(X_1) \times dom(X_2) \ldots \times dom(X_n)$.

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10

Návrh relací *- Funkční závislost*

Mějme schéma R(A), uvažujme $X \subseteq A$

Funkční závislost:

Mějme množiny atributů $B \subseteq A$, $C \subseteq A$. Říkáme, že C závisí funkčně na B (nebo B funkčně určuje C), jestliže ke každé B-hodnotě existuje **nejvýše** jedna C-hodnota.

Dané tvrzení označujeme **B → C**

C funkčně nezávisí na B: **B ↛ C**

Návrh relací - *Odvoditelnost FZ*

Pozorování: odvoditelnost **funkčních závislostí**

Př.:

c) PS $\rightarrow$ S ... platí vždy

b) platí-li $\left.\begin{array}{l} PS \rightarrow Z \\ PS \rightarrow S \end{array}\right\} \Leftrightarrow PS \rightarrow ZS$

Funkční závislosti - Armstrongova pravidla

Mějme R(A), nechť $X \subseteq A$, $Y \subseteq A$, $Z \subseteq A$

triviální funkční závislosti:
jestliže $Y \subseteq X$, pak $X \to Y$ (FZ1)
př.: $UM \to U$

tranzitivita:
jestliže $X \to Y$ a $Y \to Z$, pak $X \to Z$ (FZ2)
př.: $HS \to HM$ a $HM \to P$, pak také platí $HS \to P$.

kompozice pravé strany:
jestliže $X \to Y$ a $X \to Z$, pak $X \to YZ$ (FZ3)

dekompozice pravé strany:
jestliže $X \to YZ$, pak $X \to Y$ a $X \to Z$ (FZ4)

Návrh relací - Armstrongova pravidla

Příklad: Rozvrh(M,H,U,P,S,Z)

$P \rightarrow U$ $HM \rightarrow P$ $HU \rightarrow M$ $PS \rightarrow Z$ $HS \rightarrow M$

- Podle (FZ1) platí $HM \rightarrow H$ a $HU \rightarrow H$.
- Podle (FZ3) z $HU \rightarrow H$ a $HU \rightarrow M$ odvodíme $HU \rightarrow HM$
- Podle (FZ2) z $HM \rightarrow P$ a $P \rightarrow U$ odvodíme $HM \rightarrow U$
- Podle (FZ3) z $HM \rightarrow H$ a $HM \rightarrow U$ odvodíme $HM \rightarrow HU$

Vidíme, že HM a HU jsou **funkčně ekvivalentní**.

$HM \leftrightarrow HU$

Návrh relací - Klíč

Uzávěr množiny atributů X^+ vzhledem k F je množina všech atributů funkčně závislých na X. Označujeme jej X^+ (def. 3.4.5)

Definice klíče:

Mějme R(A), nechť $K \subseteq A$,

K je klíčem schématu R(A), jestliže splňuje dvě vlastnosti:

- $K \rightarrow A$ $K^+ = A$
- neexistuje $K' \subset K$ taková, že $K' \rightarrow A$.

Návrh relací - Armstrongova pravidla

Příklad: Rozvrh(S,H,M,U,P,Z)

F: $P \to U$, $HM \to P$, $HU \to M$, $PS \to Z$, $HS \to M$

Co je klíčem schématu?

Triviální FZ

Tranzitivita

$P^+ = \{P, U\}$

$HM^+ = \{H,M,P,U\}$

$HU^+ = \{H,U,M,P\}$

$PS^+ = \{P,S,Z,U\}$

$HS^+ = \{H,S,M,P,Z,U\}$

HS je klíčem

Normální formy schémat relací

PROGRAM	NÁZEV_K	JMÉNO_F	ADRESA	DATUM
	Blaník	Top gun	Václ.n. 4	29.03.94
	Blaník	Kmotr	Václ.n. 4	08.03.94
	Mír	Nováček	Starostrašnická 3	10.03.94
	Mír	Top gun	Starostrašnická 3	09.03.94
	Mír	Kmotr	Starostrašnická 3	08.03.94

možná ztráta informace

redundance

Integritní omezení:

- IO1: Klíčem schématu je{NÁZEV_K, JMÉNO_F}.
- IO2: Každé kino má právě jednu adresu

Normální formy schémat relací

Intuitivním řešením je dekompozice
ADRESÁŘ(NÁZEV_K,ADRESA),
PROGRAMY(NÁZEV_K, JMÉNO_F, DATUM)

PROGRAMY	NÁZEV_K	JMÉNO_F	DATUM
	Blaník	Top gun	29.03.94
	Blaník	Kmotr	08.03.94
	Mír	Nováček	10.03.94
	Mír	Top gun	09.03.94
	Mír	Kmotr	08.03.94

ADRESÁŘ	NÁZEV_K	ADRESA
	Blaník	Václ.n. 4
	Mír	Starotrašnická 3

- adresa kina je pouze jednou (odstraněna redundance)
- lze evidovat i kino, kde se (právě) nic nehraje (nehrozí ztráta informace o kinu, když bude ‘stát’)
- podstata řešení: odstraněna závislost neklíče (adresa) na pouhém podklíči (Název_k)

Normální formy schémat relací

FILM1	JMÉNO_F	HEREC	OBČANSTVÍ	ROK
	Černí baroni	Landovský	CZ	94
	Top gun	Cruise	USA	86
	Kmotr	Brando	USA	72
	Nováček	Brando	USA	90
	Vzorec	Brando	USA	80

redundance

IO1: Klíčem schématu je JMÉNO_F.
IO2: Každý herec má právě jedno občanství

- Nelze sledovat občanství herců, kteří “nehrají”,
- Ztratíme informaci o občanství herce, jesliže jeho filmyvypadnou z databáze

možná ztráta informace

Normální formy schémat relací

Intuitivním řešením je dekompozice

OSOBNÍ_ÚDAJE(HEREC, OBČANSTVÍ)

FILM2(JMÉNO_F, HEREC, ROK),

OSOBNÍ_ÚDAJE	HEREC	OBČANSTVÍ
	Landovský	CZ
	Cruise	USA
	Brando	USA

FILM2	JMÉNO_F	HEREC	ROK
	Černí baroni	Landovský	94
	Top gun	Cruise	86
	Kmotr	Brando	72
	Nováček	Brando	90
	Vzorec	Brando	80

- občanství herce je pouze jednou (odstraněna redundance)
- lze evidovat i občanství herce, jehož filmy vypadly z db (nehrozí ztráta informace o občanství herce, který “stojí”)
- podstata řešení: odstraněna závislost neklíče (občanství) na jiném neklíči (herec)

Normální formy schémat relací

V obou předchozích příkladech byly neklíčové atributy závislé na klíči. Některé z nich však nepřímo - tranzitivně.

V prvním případě šlo o tranzitivitu

klíč $\rightarrow$ podklíč $\rightarrow$ neklíč

V druhém případě šlo o tranzitivitu

klíč $\rightarrow$ neklíč $\rightarrow$ neklíč

Jsou-li všechny neklíčové atributy závislé na klíči přímo a nikoliv tranzitivně, pak je schéma ve **3NF**

Poznámka: má-li schéma více klíčů (klíč1$\leftrightarrow$klíč2),

nebude nám vadit *klíč1*$\rightarrow$*klíč2*$\rightarrow$*neklíč*

Poznámka2: Jsou-li všechny atributy schématu součástí ***nějakého*** *klíče, je schéma ve 3NF.*

Normální formy schémat relací

Mějme R(A)

Nechť $X \subset A$, $Y \subset A$ a $C \in A$, $C \notin X$ $C \notin Y$.

Nechť dále $X \rightarrow Y \rightarrow C$ a neplatí, že $Y \rightarrow X$.

Pak říkáme, že C je **tranzitivně závislý** na X.

Definice 3.4.6:
Říkáme, že schéma relace R je ve **3. normální formě** (3NF), jestliže každý neklíčový atribut schématu R není tranzitivně závislý na žádném klíči schématu.

Normální formy schémat relací – BCNF - motivace

Mějme ROZVRH(MHUP), HU→M, HM→P, P→U

lze odvodit klíče: <u>HU</u>, <u>HM</u>, <u>HP</u>

P→U ... závislost mezi dvěma podklíči

ROZVRH vyhovuje kritériu pro 3NF (Proč?)

a přeci je v datech redundance!

ROZVRH	PŘEDNÁŠKA	UČITEL	MÍSTNOST	HODINA
	Systémy	Král	S4	Po7
	Programování	Kryl	S7	Po9
	Programování	Kryl	S3	Út3

redundance

Normální formy schémat relací - BCNF

Existuje zde závislost **část_ klíče1 → část_klíče2**

P→ U

dekompozice OBS(<u>P</u>,U),
ROZVRH1(<u>HM</u>P)

- zmizela redundance v atributu U
- neztratí se informace, že Kryl přednáší Programování, když toto vypadne z rozvrhu
- řešení spočívá v odstranění závislosti části jednoho klíče na části druhého klíče

Normální formy schémat relací

Definice: 3.4.7
Říkáme, že schéma relace R je v **Boyce - Coddově normální formě** (BCNF), jestliže pro každou netriviální závislost $X \rightarrow Y$ platí, že X obsahuje klíč schématu R.

Poznámka:

Každé schéma, které je v BCNF, je také ve 3NF. Obrácené tvrzení obecně neplatí.

Má-li ale schéma jediný klíč, nebo jednoduché klíče, potom je-li ve 3NF je také v BCNF.

Normální formy schémat relací – BCNF – příklad 2

Příklad 3.4.7. Uvažujme schéma relace

ADRESÁŘ(MĚSTO, ULICE, DUM, PSČ).

F: {MĚSTO, ULICE} → PSČ, PSČ → MĚSTO

{MĚSTO,ULICE,DUM} je klíčem (→ {PSČ,MĚSTO,ULICE,DUM})

{PSČ,ULICE,DUM} je klíčem (→ {PSČ,MĚSTO,ULICE,DUM})

Schéma *nemá žádný neklíčový atribut* a je tedy ve 3NF. Nikoliv však v BCNF.

ADRESÁŘ lze nahradit dekompozicí.

dekompozice1:

A1(PSČ, MĚSTO)

B1(PSČ, ULICE, DUM)

dekompozice2:

A2(MĚSTO,ULICE,PSČ)

B2(MĚSTO,ULICE,DUM)

Úprava relačního schématu databáze

NORMALIZACE

Eliminaci aktualizačních anomálií zajišťujeme převedením relačního schématu do 3NF, resp. BCNF.

(Normalizovat lze pomocí) DEKOMPOZICE

Původní schéma: $R(U, F)$

Dekomponované schéma: $\{ R_i (U_i,F_i) \}^n_{i=1}$ kde $\cup U_i = U$

Kvalita dekompozice (požadavky):

P1: Výsledná schémata by měla mít "stejnou" sémantiku.

P2: Nové relace by měly obsahovat "stejná" data, jaká by obsahovala původní relace.

Pokrytí původní množiny závislostí F (P1)

Cílem bude, aby původní schéma a schémata získaná dekompozicí nějak odrážela stejné závislosti.

$$\mathbf{F^{+} = (\cup F_{i})^{+}}$$

zpět k příkladu: ADRESÁŘ(MĚSTO, ULICE, DUM, PSČ).
F: {MĚSTO, ULICE} → PSČ, PSČ → MĚSTO

Dekompozice: SEZNAM_POŠT(<u>PSČ</u>, MĚSTO)
POŠTOVNÍ_RAJON(<u>PSČ, ULICE, DUM</u>)

Ve schématu SEZNAM_POŠT lze kontrolovat původní funkční závislost PSČ → MĚSTO.

Původní závislost {MĚSTO, ULICE} → PSČ pokryta není.

Pokrytí původní množiny závislostí F

Příklad 3.4.7.

FILM1(JMÉNO_F, ROK, HEREC, PŘÍSLUŠNOST)
F: HEREC → PŘÍSLUŠNOST, JMÉNO_F → HEREC,
JMÉNO_F → PŘÍSLUŠNOST

Dekompozice podle HEREC → PŘÍSLUŠNOST:

OSOBNÍ_ÚDAJE(HEREC, PŘÍSLUŠNOST), HEREC → PŘÍSLUŠNOST

FILM2(JMÉNO_F, ROK, HEREC), JMÉNO_F→ HEREC

Závislost JMÉNO_F → PŘÍSLUŠNOST je pokryta, protože je **odvoditelná** ze závislostí, které platí na schématech OSOBNÍ_ÚDAJE a FILM2.

Pokrytí původní množiny závislostí F

Definice 3.4.8:
Mějme schéma databáze

$$R = \{S(A,F)\}$$

a dekompozici
$\mathbf{R_D} = \{(R_i(A_i),F_i),\ 1 \leq i \leq n,\ n \geq 1\}$.

Řekneme, že $\mathbf{R_D}$ má vlastnost pokrytí závislostí, jestliže

$$\mathbf{F^+ = \left(\bigcup_{i=1}^{n} F_i\right)^+}$$

Pokrytí původní množiny závislostí F

Příklad 3.4.7.

JIZDA(C_AUTA, RIDIC, TYP,OBSAH_M)

F: C_AUTA → TYP .... nevyhovuje 2NF

TYP → OBSAH_M .... nevyhovuje 3NF

žádná FZ není pokryta

Dekompozice1:
R1(TYP,RIDIC),
R2(C_AUTA,RIDIC,OBSAH_M)

2. původní FZ není pokryta

Dekompozice2 („podle 1. FZ"):
R1(C_AUTA,TYP) **,C_AUTA→TYP**
R2(C_AUTA, RIDIC, OBSAH_M)

obě původní FZ jsou pokryty

Dekompozice3 (začneme „podle 2. FZ"):
R1(TYP, OBSAH_M) **, TYP → OBSAH_M**
R2(C_AUTA, RIDIC, TYP) **,C_AUTA→TYP**

Bezztrátové spojení (P2)

Nové relace by měly obsahovat "stejná" data, jaká by obsahovala původní relace.

Dekompozici schématu lze považovat za několik **projekcí** původní relace na množiny atributů nových schémat. Kvalitní dekompozice bude taková, která bude mít vlastnost zpětného **bezztrátového spojení.**

Pro každou přípustnou relaci S* by mělo platit

$$\mathbf{S^* = \mathop{\ast}_{i=1}^{n} S_i^*[A_i]}$$

*Příklad špatné dekompozice – **není bezztrátová!***

Z	PŘEDN	STUD	ZN
	Programování	Novák	2
	Programování	Volák	3
	Systémy	Zíka	1
	Systémy	Tupý	2
	Systémy	Novák	2
	Systémy	Bílý	1

Proveďme dekompozici Z*
ZAPIS(PŘEDN, STUD)
HODN (PŘEDN, ZN)

ZAPIS*:=Z* [PŘEDN, STUD]

HODN:=Z*[PŘEDN, ZN]

ZAPIS

Z	PŘEDN	STUD
	Programování	Novák
	Programování	Volák
	Systémy	Zíka
	Systémy	Tupý
	Systémy	Bílý
	Systémy	Novák

HODN

Z2	PŘEDN	ZN
	Programování	2
	Programování	3
	Systémy	1
	Systémy	2
	Systémy	1

Zpětné spojení bude "větší" než původní Z*.
Bude např. obsahovat n-tici **(Programování, Novák, 3).**
Zpětné spojení není bezztrátové.

Přestože obdržíme více n-tic, informace je méně, nevíme, co platí a co ne.

Bezztrátové spojení (P2)

Tvrzení 3.4.6:
Nechť S(A) je schéma relace a $\{S_i(A_i)\}$, i $\in$<1,n>, n>1, určuje jeho dekompozici. Pak pro každou relaci S* platí

Aby platila rovnost, je třeba provést dekompozici dostatečně smysluplně.

$$\mathbf{S^* \subseteq \mathop{*}_{i=1}^{n} S^*_i\,[A_i]}$$

Tvrzení 3.4.7. Mějme schéma **R(A,B,C)**, kde A,B,C jsou disjunktní množiny atributů, a funkční závislost **B $\rightarrow$ C**. Rozložíme-li R na schémata **R1($\underline{\text{B}}$,C)** a **R2(A,B)**, je takto provedená dekompozice **bezztrátová**.

Naopak, je-li dekompozice R1(B,C) a R2(A,B) bezztrátová, musí platit buď B $\rightarrow$ C nebo B $\rightarrow$ A.

Bezztrátové dekompozice - příklad

Dekompozice může mít vlastnost bezztrátového spojení,

nemusí však mít vlastnost pokrytí závislostí.

ADRESÁŘ(MĚSTO, <u>ULICE, DUM, PSČ</u>).

F: (MĚSTO,ULICE) →PSČ , PSČ→ MĚSTO

Dekompozice (bezztrátová) :

SEZNAM_POŠT(<u>PSČ</u>, MĚSTO) PSČ→ MĚSTO (platí dále)

POŠTOVNÍ_RAJON(<u>PSČ, ULICE, DUM</u>)

{MĚSTO, ULICE} →PSČ (tuto FZ jsme ztratili)

Bezztrátová dekompozice - úvaha

Má-li dekompozice vlastnost pokrytí závislostí,

nemusí být bezztrátová.

Například:

R($\underline{A,C}$,B,D) $\Rightarrow$ R1($\underline{A}$,B), $A \rightarrow B$

F: $A \rightarrow B$, $C \rightarrow D$ R2($\underline{C}$,D), $C \rightarrow D$

$(R1 * R2)^* \neq R^*$

jak to napravíme? R3($\underline{A,C}$)

$(R3 * R1 * R2)^* = R^*$

Pokrytí závislostí a bezztrátové spojení

Důsledek porušení “pokrytí závislostí” (P1):

CHUDŠÍ SÉMANTIKA

Důsledek porušení “bezeztrátovosti“ (P2):

NEJDE O STEJNÁ DATA

Algoritmus dekompozice

- Předpoklad schématu **univerzální relace**
  - jednoznačnost jmen atributů,
  - atribut hraje pouze jednu roli

    Příklad: jméno ZNÁMKA je vyhrazeno pro atribut „hodnocení studenta u zkoušky“ a nemůže být současně použito pro atribut „ohodnocení kvalifikace učitele“
- Předpoklad **jednoznačnosti vztahů mezi atributy**

  Příklad:
  VEDOUCÍ_PROJEKTU(UČITEL, STUDENT , PROJEKT)
  VYUKA(UČITEL, STUDENT, PŘEDMĚT)

Algoritmus dekompozice

- postup:
  - Schémata se rozkládají binárně dle tvrzení o bezeztrátové dekompozici
  - Strategie: “nalomit” tranzitivitu, dekomponovat podle FZ, která způsobuje, že schéma není v 3NF.
  - Každé nové schéma se testuje na 3NF
- výsledek:
  - je ve 3NF
  - je zachována bezeztrátovost
  - obecně nejsou pokryty závislosti

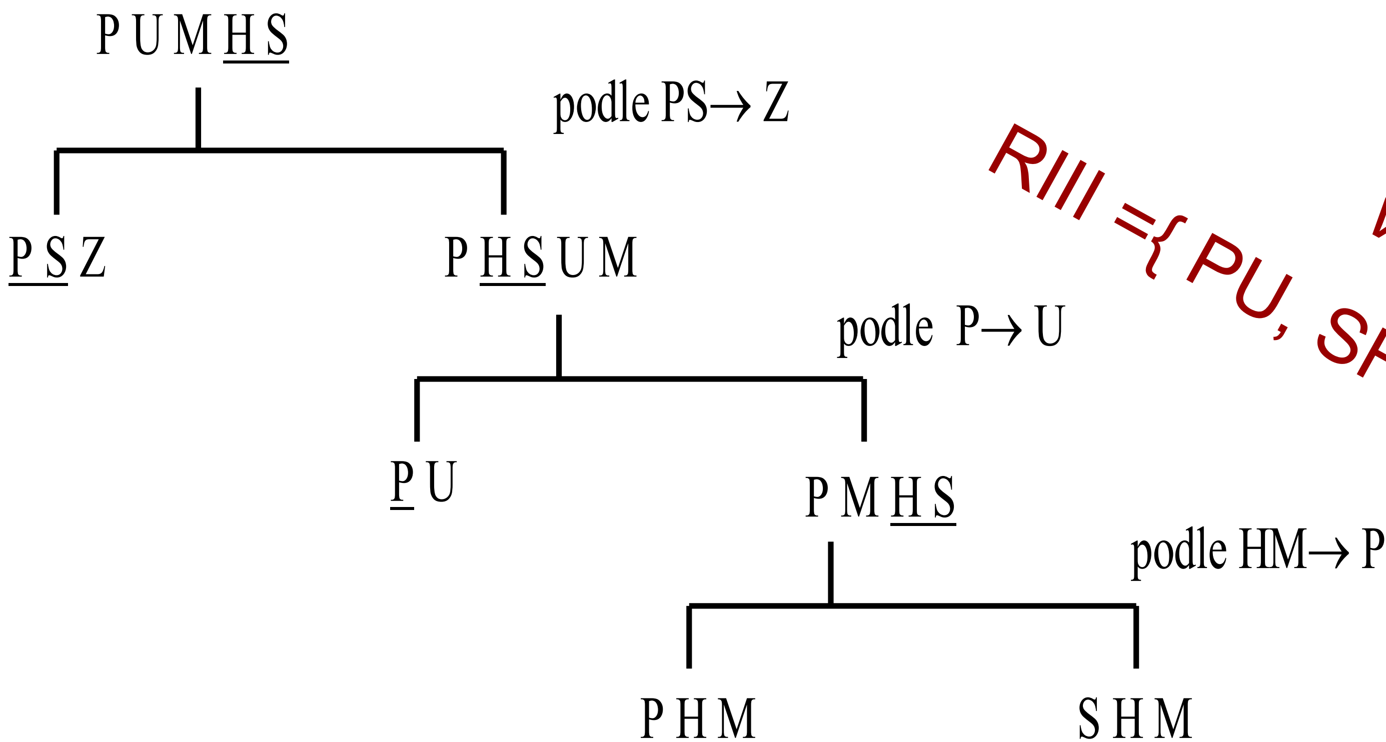
R(<u>K</u>, C, D), C $\rightarrow$ D

$\Downarrow$

R1(<u>C</u>, D), R2(<u>K</u>,C)

Algoritmus dekompozice – řešení 1

Příklad univerzita: F: P→ U, HM→ P, HU→ M, PS→ Z, HS→ M

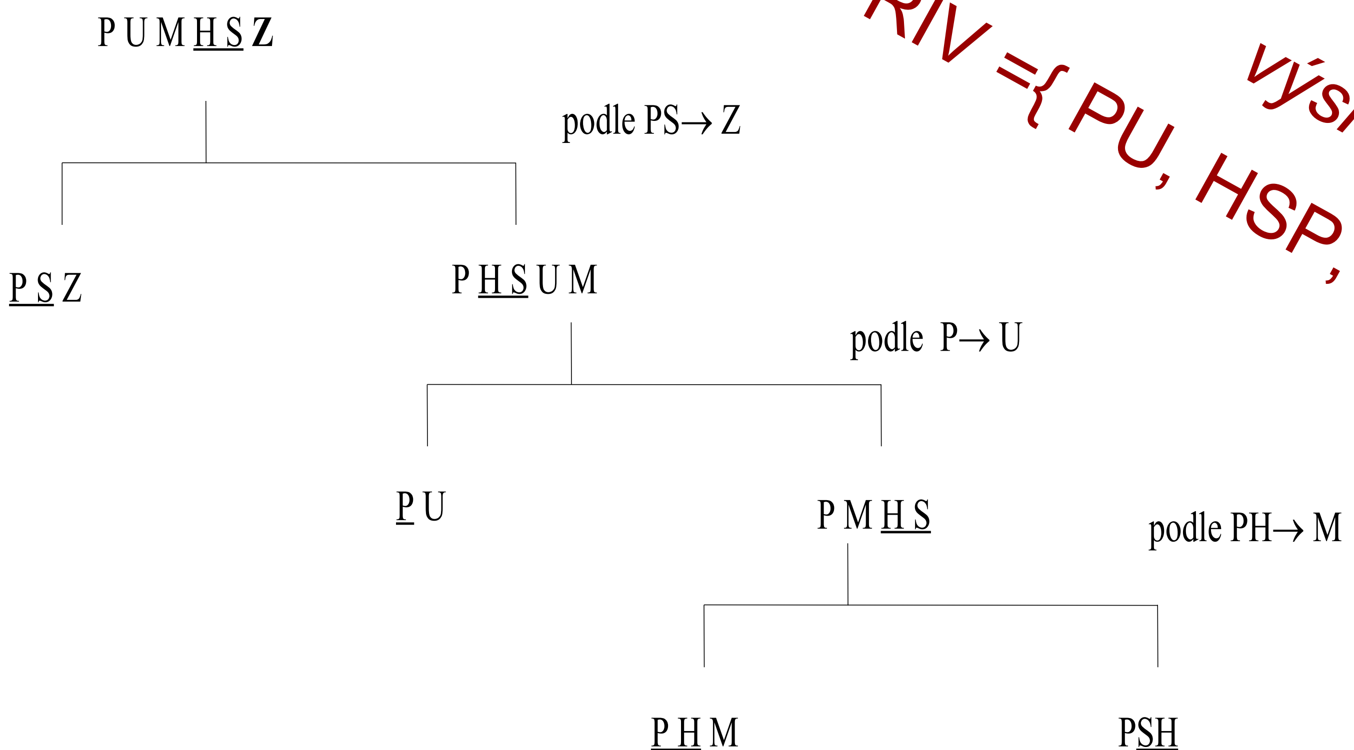


výsledek

RIII ={ PU, SHM, PSZ, PHM }

Algoritmus dekompozice – řešení 2

Příklad univerzita: F: P→ U, HM→ P, HU→ M, PS→ Z, HS→ M
Dekompozice do BCNF varianta 2



výsledek

RIV ={ PU, HSP, PSZ, PHM }

PH→ M je odvoditelná z **F**

Úprava množiny FZ - pojmy

V Algoritmu normalizace jsme mlčky předpokládali, že empiricky zjištěná F je v prvním kroku konsolidovaná.

Chtěli bychom nějaké „rozumné“ pokrytí F – jedním takovým je **minimální pokrytí.**

Závislost, která má na pravé straně jeden atribut, nazýváme **elementární** funkční závislost.

Množina všech funkčních závislostí odvoditelných z F se nazývá **uzávěr F** (definice 3.4.2)
Značíme: F^+.

Úprava množiny funkčních závislostí - pojmy

Pokrytí množiny funkčních závislostí F je množina funkčních závislostí G, taková, že $G^+ = F^+$ (def. 3.4.3)

Je-li F' množina elementárních závislostí, která vznikne z F dekompozicí jejích neelementárních závislostí, platí $F^+ = F'^+$.
Vzniklo tak **kanonické pokrytí**

Závislost f je **redundantní v F**, jestliže je odvoditelná ze zbytku F

$$(F - \{ f \})^+ = F^+$$

Odstraněním všech redundancí vznikne **neredundantní pokrytí F** (def. 3.4.4)

Nereduntnantních pokrytí může být více.

Úprava množiny funkčních závislostí - motivace

Příklad: První den jsme empiricky odhalili $F = \{AC \rightarrow B, BC \rightarrow D\}$
Druhý den odhalíme $AC \rightarrow D$
Ověřme, zda přináší novou informaci.

1. $AC \rightarrow B$
2. $AC \rightarrow C$

$\left.\begin{array}{l} AC \rightarrow B \\ AC \rightarrow C \end{array}\right\} \; AC \rightarrow BC$

$\left.\begin{array}{l} AC \rightarrow BC \\ BC \rightarrow D \end{array}\right\} \; AC \rightarrow D$

3. $BC \rightarrow D$

$\{AC \rightarrow D\} \in F^+$
Přidáním by vznikla redundance

Úloha: Zjistit, zda f je redundantní v F, tj. zda $(F - \{ f \})^+ = F^+$

- Neredundantní pokrytí není dáno jednoznačně
- Nered. pokrytí nemusí být podmnožinou F, může vzniknout z F^+

Úprava množiny funkčních závislostí

Uzávěr množiny atributů X^+ vzhledem k F je množina všech atributů funkčně závislých na X. Označujeme jej X^+ (def. 3.4.5)

Obsahuje-li F závislost $X \rightarrow Y$ a existuje atribut $A \in X$ takový, že platí $(X-A)^+ = X^+$, říkáme, že A je na levé straně dané závislosti **redundantním atributem**

Př.: $AB \rightarrow Y \in F$; jestliže $B^+_F = AB^+_F$, potom A je redundantní

Závislost, u které neexistují na levé straně žádné redundantní atributy se nazývá **redukovaná závislost**

Úprava množiny FZ – konstrukce min. pokr. příklad

Příklad :

F: AB→C, C→A, BC→D, ACD→B, **D→EG**, BE→C, **CG→BD**, **CE→AG**

⇓ ***do kanonického tvaru***

F': AB→C, C→A, BC→D, ~~A~~CD→B, **D→E, D→G**, BE→C, **CG→B,**
CG→D, **CE→A, CE→G**

⇓ ***redukce redundantních atributů***

F'': AB→C, C→A, BC→D, CD→B, **D→E, D→G**, BE→C, ~~**CG→B**~~,
CG→D, ~~**CE→A**~~, **CE→G**

⇓ ***redukce redundantních FZ***

F''': AB→C,C→A,BC→D,CD→B, **D→E, D→G**, BE→C, **CG→D**, CE→G

Úprava množiny FZ – konstrukce min. pokr.

Algoritmus 3.4.3. Nalezení **minimálního pokrytí** pro množinu funkčních závislostí.

Vstup: F nad množinou atributů A relace R(A)

Výstup: minimální pokrytí G

begin 1. Dekomponuj pravé strany funkčních závislostí, tedy převeď FZ do **elementárního tvaru**, sestroj pro F **kanonické pokrytí** F'.

2. Odstraň redundantní atributy, tedy uprav F' na F'' tak, aby všechny f byly **redukované**.

3. Odstraň redundantní funkční závislosti, tedy pro F'' vytvoř **neredundantní pokrytí** F'''

end

Úprava množiny funkčních závislostí

Algoritmus 3.4.2. Nalezení **neredundantního pokrytí** pro množinu elementárních funkčních závislostí F'.

Vstup: F' nad množinou atributů A relace R(A)

Výstup: neredundantní pokrytí G

begin G := F'

for each f $\in$ G do

if **f** $\in$ **(G - { f })**$^+$ then G := G - { f }

end

Problém příslušnosti f do F

Úprava množiny FZ – konstrukce min. pokr. - příklad 2

Příklad 3.4.3. Nechť je dáno schéma
R(A,B,C,D) a F = {A →AC, B→ABC, D→ABC}
Spočti minimální pokrytí

Krok 1: převedení na elementární závislosti:

F'= {~~A → A~~, A → C, B→A, ~~B→B~~, ~~B→C~~, ~~D→A~~, D→B, ~~D→C~~}

Krok 2: redukce levých stran:

Všechny závislosti jsou již redukované

Krok 3: eliminace redundantních FZ

A→A, B →B jsou triviální

A→C ?

B→A ?

B→C ? *(platí B→A a A → C, tedy B→C ∈ F⁺)*

D→A ? *(když ji vyřadíme, stále bude platit D+= DABC)*

D→B ?

D→C ?

Výsledné neredundantní pokrytí G pro F' (minimální) je

G'' = {A → C, B→A, D→B} **Co je klíč?** **Je R(A,B,C,D),G'' ve 3NF?**