Úvod do databázových systémov

http://www.dcs.fmph.uniba.sk/~plachetk

/TEACHING/DB1

Tomáš Plachetka

Fakulta matematiky, fyziky a informatiky,
Univerzita Komenského, Bratislava

Zima 2021–2022

Dekompozícia zachovávajúca funkčné závislosti

Definícia. Relačná schéma (r, F) je relácia r spolu s množinou funkčných závislostí F, ktoré platia v r

(Množina F indukuje ďalšie funkčné závislosti platné v r.)

Definícia. Dekompozícia $(r_1, F_1), ..., (r_n, F_n)$ **zachováva funkčné závislosti** schémy (r, F), ak každá platná funkčná závislosť $\mathbf{X} \rightarrow \mathbf{Y}$ z F je v uzávere platných funkčných závislostí z F_i, t.j. $F^+ = (\cup F_i)^+$

Aby sme dekompozíciu vôbec považovali za rozumnú, tak sa musí **spájať bezstratovo**. V záujme jednoduchosti automatickej kontroly integrity dát (constraints) je tiež žiadúce, aby zachovávala funkčné závislosti pôvodnej relačnej schémy

Test bezstratovosti rozkladu pre dve relácie

Veta. Dekompozícia do 2 relácií je bezstratová ak množina spoločných atribútov je nadkľúč v r_1 alebo v r_2

Algoritmus testovania bezstratovosti rozkladu (polynomiálna časová zložitosť):

- Over, či platí v r funkčná závislosť $r_1 \cap r_2 \rightarrow r_1$ alebo $r_1 \cap r_2 \rightarrow r_2$. Ak platí aspoň jedna z nich, tak je dekompozícia bezstratová, inak je stratová

Pozor! Tento test nefunguje pre dekompozíciu do 3 či viac relácií!

Pre dekompozíciu do 3 a viac relácií treba použiť komplikovanejší test (chase), avšak stále s polynomiálnou časovou zložitosťou

Test bezstratovosti rozkladu pre N relácií (chase)

Input: Dekompozícia schémy r, F do r_1, ..., r_m

1. Vyrob maticu S s 1 riadkom pre každú podreláciu r_1, ..., r_m

a s 1 stĺpcom pre každý atribút r

2. Nastav počiatočné hodnoty, S[i, j] := „$b_{i,j}$“

3. for (i = 1...m)

    for (j = 1...n)

        if (atribút A_j patrí do r_i) then S[i, j] := „a_j“

4. Opakuj kým matica S mení:

    for all ($\mathbf{X} \rightarrow \mathbf{Y} \in F$)

        for (všetky dvojice riadkov s rovnakými symbolmi v tých stĺpcoch, ktoré zodpovedajú atribútom v **X**)

            Zjednoť riadky v tých 2 riadkoch, pritom preferuj symboly a_*

5. Ak jeden riadok obsahuje len symboly „a_*“, rozklad je bezstratový, inak nie

Test bezstratovosti rozkladu: príklad

Príklad: r(A, B, C, D, E, F, G, H)

A → B, ABCD → E, EF → GH, ACDF → EG

Dekompozícia:

r_1(AB), r_2(ACDE), r_3(EFG), r_4(EFH).

Spája sa táto dekompozícia bezstratovo?

Keďže ide o dekompozíciu do viac ako 2 tabuliek, treba použiť tabuľkovú metódu (tzv. chase)

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
AB	**a1**	**a2**	b13	b14	b15	b16	b17	b18
ACDE	**a1**	b22	**a3**	**a4**	**a5**	b26	b27	b28
EFG	b31	b32	b33	b34	**a5**	**a6**	**a7**	b38
EFH	b41	b42	b43	b44	**a5**	**a6**	b47	**a8**

Test bezstratovosti rozkladu: príklad, pokračovanie

$A \rightarrow B$

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
a1	**a2**	b13	b14	b15	b16	b17	b18
a1	**a2**	a3	a4	a5	b26	b27	b28
b31	b32	b33	b34	a5	a6	a7	b38
b41	b42	b43	b44	a5	a6	b47	a8

$ABCD \rightarrow E$

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
a1	a2	b13	b14	b15	b16	b17	b18
a1	a2	a3	a4	a5	b26	b27	b28
b31	b32	b33	b34	a5	a6	a7	b38
b41	b42	b43	b44	a5	a6	b47	a8

Test bezstratovosti rozkladu: príklad, pokračovanie

EF → GH

A	B	C	D	E	F	G	H
a1	a2	b13	b14	b15	b16	b17	b18
a1	a2	a3	a4	a5	b26	b27	b28
b31	b32	b33	b34	**a5**	**a6**	**a7**	**a8**
b41	b42	b43	b44	**a5**	**a6**	**a7**	**a8**

ACDF → EG

A	B	C	D	E	F	G	H
a1	a2	b13	b14	b15	b16	b17	b18
a1	a2	a3	a4	a5	b26	b27	b28
b31	b32	b33	b34	a5	a6	a7	a8
b41	b42	b43	b44	a5	a6	a7	a8

Po aplikácii ľubovoľnej funkčnej závislosti sa tabuľka už nezmení.
Dekompozícia AB, ACDE, EFG, EFH **sa nespája bezstratovo**

Test bezstratovosti rozkladu: iný príklad

Iný príklad: r(A, B, C, D, E, F, G, H)

A → B, ABCD → E, EF → GH, ACDF → EG

Dekompozícia:

r_1(AB), r_2(ACDE), r_3(EFG), r_4(EFH), **r_5(ACDF).**

Spája sa táto dekompozícia bezstratovo?

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
AB	**a1**	**a2**	b13	b14	b15	b16	b17	b18
ACDE	**a1**	b22	**a3**	**a4**	**a5**	b26	b27	b28
EFG	b31	b32	b33	b34	**a5**	**a6**	**a7**	b38
EFH	b41	b42	b43	b44	**a5**	**a6**	b47	**a8**
ACDF	**a1**	b52	**a3**	**a4**	b55	**a6**	b57	b58

Test bezstratovosti rozkladu: iný príklad, pokračovanie

$A \rightarrow B$

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
a1	**a2**	b13	b14	b15	b16	b17	b18
a1	**a2**	a3	a4	a5	b26	b27	b28
b31	b32	b33	b34	a5	a6	a7	b38
b41	b42	b43	b44	a5	a6	b47	a8
a1	**a2**	a3	a4	b55	a6	b57	b58

$ABCD \rightarrow E$

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**
a1	a2	b13	b14	b15	b16	b17	b18
a1	**a2**	**a3**	**a4**	**a5**	b26	b27	b28
b31	b32	b33	b34	a5	a6	a7	b38
b41	b42	b43	b44	a5	a6	b47	a8
a1	**a2**	**a3**	**a4**	**a5**	a6	b57	b58

Test bezstratovosti rozkladu: iný príklad, pokračovanie

EF → GH

A	B	C	D	E	F	G	H
a1	a2	b13	b14	b15	b16	b17	b18
a1	a2	a3	a4	a5	b26	b27	b28
b31	b32	b33	b34	**a5**	**a6**	**a7**	**a8**
b41	b42	b43	b44	**a5**	**a6**	**a7**	**a8**
a1	a2	a3	a4	**a5**	**a6**	**a7**	**a8**

Dekompozícia AB, ACDE, EFG, EFH, ACDF

sa spája bezstratovo, lebo posledný riadok obsahuje iba symboly a*

Prvá normálna forma (1NF)

Definícia. Relačná schéma (r, F) je v prvej normálnej forme (1NF), ak žiaden z atribútov r nie je zložený atribút a databáza neobsahuje duplikáty

Inak povedané: databáza je bez duplikátov

a dáta nemajú štruktúru

Príklad schémy, ktorá nie je v 1NF kvôli štruktúrovaným dátam (weekday)

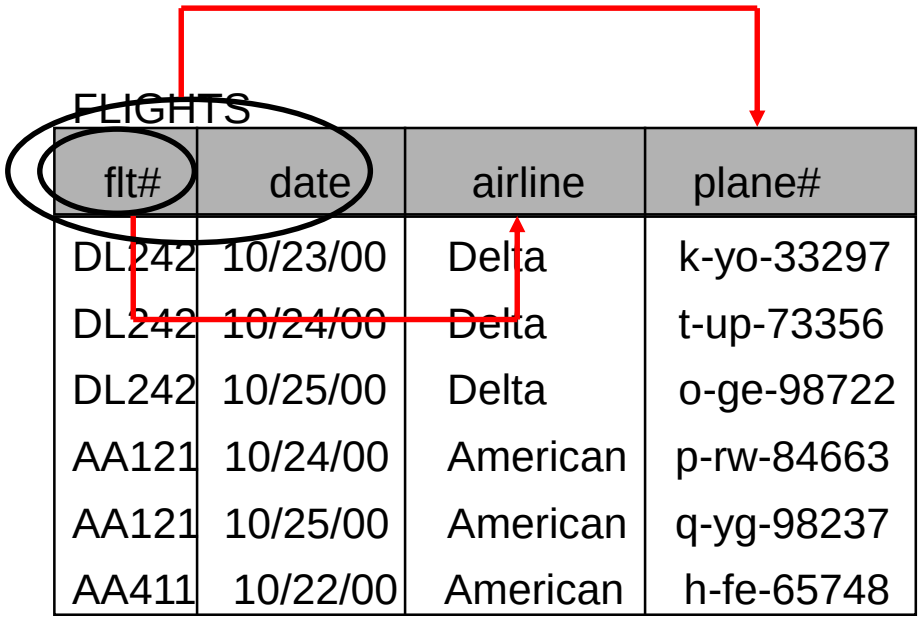
FLT-SCHEDULE

flt#	weekday	airline	dtime	from	atime	to
DL242	MO WE FR	DELTA	10:40	ATL	12:30	BOS
SK912	SA SU	SAS	12:00	CPH	15:30	JFK
AA242	MO FR	AA	08:00	CHI	10:10	ATL

Druhá normálna forma (2NF)

Definícia. Relačná schéma (r, F) je v druhej normálnej forme (2NF), ak je v 1NF a ak v nej neexistuje platná funkčná závislosť **X** $\rightarrow$ Y, kde **X** je striktná podmnožina nejakého kľúča a Y nepatrí do žiadneho kľúča (Y je jeden atribút)

Príklad schémy, ktorá nie je v 2NF: Jediný kľúč je dvojica [flt#, date], ale airline funkčne závisí len od flt#.



flt#	date	airline	plane#
DL242	10/23/00	Delta	k-yo-33297
DL242	10/24/00	Delta	t-up-73356
DL242	10/25/00	Delta	o-ge-98722
AA121	10/24/00	American	p-rw-84663
AA121	10/25/00	American	q-yg-98237
AA411	10/22/00	American	h-fe-65748

Tretia normálna forma (3NF)

Definícia. Relačná schéma (r, F) je v tretej normálnej forme (3NF), ak pre každú platnú netriviálnu (takú, že ľavá strana sa nedá skrátiť) funkčnú závislosť $\mathbf{X} \rightarrow Y$ platí, že buď **X** je nadkľúč v r alebo Y je časťou nejakého kľúča v r (Y je jeden atribút)

Platí:

- Každá binárna relácia je v 3NF
- Ak r nie je v 3NF, tak existujú atribúty A a B také, že
  r - AB $\rightarrow$ A (toto sa dá priamo použiť pre dekompozíciu do 3NF)

Tretia normálna forma (3NF)

Testovanie, či je schéma v 3NF je NP-ťažké. Avšak rozkladanie schémy do nejakej 3NF, ktoré je bezstratové a zachováva všetky funkčné závislosti, má polynomiálnu časovú zložitosť (vzhľadom na počet atribútov a počet závislostí relačnej schémy).

Daňou za redukciu časovej zložitosti je zbytočná resp. „neprirodzená“ dekompozícia tabuliek.

Príklad schémy, ktorá nie je v 3NF:
planetype funkčne závisí od plane#, ale plane# nie je nadkľúč, a zároveň planetype nie je časťou žiadneho kľúča (jediný kľúč je ticket#)

TICKET-PLANE-PLANETYPE

ticket#	plane#	planetype
DL001	k-yo-33297	A320
DL002	k-yo-33297	A320
DL003	o-ge-98722	LC4
AA001	k-yo-33297	A320
AA002	q-yg-98237	B747
AA003	q-yg-98237	B747

Boyce-Coddova normálna forma (BCNF)

Definícia. Relačná schéma (r, F) je v Boyce-Coddovej normálnej forme (BCNF), ak pre každú platnú netriviálnu funkčnú závislosť $\mathbf{X} \rightarrow \mathbf{Y}$ platí, že **X** je nadkľúč v r

Platí:

- Každá binárna relácia je v BCNF
- Ak r nie je v BCNF, tak existujú atribúty A a B také, že

  $r - AB \rightarrow A$

  (toto sa dá priamo použiť pre dekompozíciu do BCNF)

Boyce-Coddova normálna forma (BCNF)

Testovanie, či je schéma v BCNF, je NP-ťažké. Bezstratový rozklad do nejakej BCNF sa dá nájsť v polynomiálnom čase s použitím nasledujúceho algoritmu, ale negarantuje zachovanie všetkých funkčných závislostí

Je NP-ťažké rozhodnúť, či vôbec existuje nejaká BCNF dekompozícia, ktorá zachováva všetky funkčné závislosti

Relačná schéma, ktorá je v BCNF, neobsahuje žiadnu redundanciu vzhľadom na funkčné závislosti.

Naivná dekompozícia do 3NF, resp. BCNF

1. Nájdi funkčnú závislosť z F^+, ktorá porušuje podmienku *NF
2. Minimalizuj jej ľavú stranu
3. Maximalizuj jej pravú stranu (nie je nutné)
4. Nech výsledok je $\mathbf{X} \rightarrow \mathbf{Y}$
5. Vytvor dve nové relácie: r - **Y** a **XY**
6. So vzniknutými reláciami proces opakuj, až kým všetky nie sú v požadovanej *NF

Tento algoritmus má pre 3NF a BCNF **exponenciálnu časovú zložitosť** (vzhľadom na počet atribútov a počet funkčných závislostí relačnej schémy)

Iná naivná dekompozícia do 3NF, resp. BCNF

1. Nájdi atribúty A a B také, že r - AB $\rightarrow$ A (ak také dva atribúty neexistujú, tak schéma je v 3NF aj v BCNF)
2. Vytvor dve nové relácie: r - B a r - A
3. So vzniknutými reláciami proces opakuj

Tento algoritmus má pre 3NF a BCNF **polynomiálnu časovú zložitosť** (vzhľadom na počet atribútov a počet závislostí relačnej schémy), ale niekedy **zbytočne dekomponuje** aj reláciu, ktorá už je v 3NF resp. BCNF

Ak schéma nie je v 3NF resp. BCNF, tak v kroku 1 sa toto zistí v polynomiálnom čase: vtedy neexistujú atribúty A a B také, že r - AB $\rightarrow$ A. Keď **sa v kroku 1 nájdu také dva atribúty A a B, že r - AB $\rightarrow$ A, tak (bohužiaľ) stále nevieme rozhodnúť**, či schéma je alebo nie je v 3NF resp. BCNF

Dekompozícia do 3NF zachovávajúca funkčné závislosti

Vstup: Relačná schéma r a **minimálne pokrytie** závislostí F

Výstup: Relačné schémy bestratovej dekompozície do 3NF, pričom všetky funkčné závislosti ostanú zachované

Dekompozícia do 3NF zachovávajúca funkčné závislosti:

1. Ak F obsahuje funkčnú závislosť, ktorá obsahuje všetky atribúty r, potom r je už v 3NF; inak každej funkčnej závislosti v F zodpovedá jedna relačná schéma

2. Zlúč relačné schémy pre funkčné závislosti s rovnakou ľavou stranou

3. Ak je niektorá podschéma podmnožinou inej, tak tú menšiu vynechaj

4. Ak žiadna z tých relačných podschém neobsahuje kľúč r, pridaj k dekompozícii schému s nejakým kľúčom. (Tento krok nevyžaduje výpočet všetkých kľúčov.)

Dekompozícia do 3NF zachovávajúca funkčné závislosti

Príklad:
V relácii r(A, B, C, D, E, F, G, H) platia funkčné závislosti
A → B, ABCD → E, EF → GH, ACDF → EG.

Minimálne pokrytie: A → B, ACD → E, EF → G, EF → H.

r nie je v 3NF, lebo platí A → B a A nie je nadkľúč ani B nie je časťou žiadneho kľúča (jediným kľúčom je ACDF).

Bezstratová dekompozícia do 3NF zachovávajúca funkčné závislosti:

r_1={AB} /* zachováva A → B */

r_2={ACDE} /* zachováva ACD → E */

r_3={EFGH} /* zachováva EF → G a EF → H */

r_4={ACDF} /* jediný kľúč r je ACDF, nie je obsiahnutý v r_1, r_2 ani r_3 */

Dekompozícia do BCNF

Pri dekompozícii sa treba snažiť zachovať funkčné závislosti (i keď nie vždy sa to dá), kvôli automatizácii integrity dát

Algoritmus dekompozície relácie R do BCNF (je prirodzené začať s dobrou 3NF dekompozíciou):

1. Dekomponuj r do 3NF so zachovaním funkčných závislostí
2. Over, či každá relácia dekompozície je v BCNF. Ak nie, nájdi v nej funkčnú závislosť $\mathbf{X} \rightarrow \mathbf{Y}$ ktorá porušuje BCNF a dekomponuj reláciu r_i do dvoch relácií, r_i - **Y** a **XY**
3. Opakuj overovanie a rozkladanie až kým sú všetky relácie v BCNF

Dekompozícia do BCNF

Príklad:

V relácii r(A, B, C, D, E, F, G, H) platia funkčné závislosti

A → B, ABCD → E, EF → GH, ACDF → EG.

Minimálne pokrytie: A → B, ACD → E, EF → G, EF → H.

Bezstratová dekompozícia do 3NF zachovávajúca funkčné závislosti:

r_1={AB}, r_2={ACDE} , r_3={EFGH}, r_4={ACDF}

Pre overenie BCNF treba najskôr vypočítať *všetky* netriviálne funkčné závislosti (ktorých ľavé strany neobsahujú zbytočné atribúty), ktoré platia v r a ktorých ľavé strany nie sú nadkľúčom v r:

A → B, ACD → E, EF → G, EF → H.

Ľavé strany všetkých týchto funkčných závislostí z r sú nadkľúčmi v r_1, r_2, r_3, r_4, preto je tá dekompozícia nielen v 3NF, ale aj v BCNF

Porušenie BCNF vs. zlomenie funkčných závislostí

Príklad:

adresy(Mesto, Ulica_Číslo, PSČ)

PSČ $\rightarrow$ Mesto

Mesto, Ulica_Číslo $\rightarrow$ PSČ

Táto schéma nie je v BCNF, lebo platí PSČ $\rightarrow$ Mesto, pričom PSČ nie je nadkľúč. Dekompozícia do BCNF

(Mesto, PSČ), (Ulica_Číslo, PSČ) láme funkčnú závislosť

Mesto, Ulica_Číslo $\rightarrow$ PSČ

Lenže pozor, tá prvá funkčná závislosť je pochybná (nie je garantované, že platí). **V tomto prípade je rozumné nerobiť žiadnu dekompozíciu**

Porušenie BCNF vs. zlomenie funkčných závislostí

Ešte jeden príklad schémy v 3NF, ale nie v BCNF (Elmasri, Navathe):

Student, Course → Instructor, Instructor → Course

TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

Figure 10.13
A relation TEACH that is in 3NF but not BCNF.

Rozumná BCNF dekompozícia *za predpokladu, že tá druhá závislosť skutočne platí*:

(Student, Instructor), (Instructor, Course)

Ešte jeden príklad rozkladu do 3NF a BCNF

r(A, B, C, D, E, F, G, H)

F→A, A→E, E→B, E→D, D→H, BG→F, CD→A, BD→E

Toto je už minimálne pokrytie.

Kľúče v r sú CGA, CGB, CGD, CGE, CGF.

3NF rozklad z tohto min. pokrytia:

FA, AE, DH, BGF, CDA, BDE, CGA

Iný 3NF rozklad (všetky atribúty okrem H sú kľúčové, takže môžu byť v jednej tabuľke):

ABCDEFG, DH

BCNF rozklad (pozor! platí F→B a tiež A→D):

FA, AE, DH, GF, BF, CA, AD, BDE, CGA

(niektoré funkčné závislosti treba rozbiť)

Vyššie normálne formy

Vyššie normálne formy (4NF, ...), odstraňujú redundanciu vzhľadom (aj) na **iné závislosti než funkčné**

Konštrukcia 4NF a vyšších má exponenciálnu časovú zložitosť

Proces normalizácie konverguje k rozkladu do binárnych relácií, čo zvyčajne nechceme

Rozhodnutie kedy normalizáciu zastaviť, vyžaduje dlhodobú prax. Niekedy je dokonca vhodné databázu denormalizovať, t.j. spojiť niektoré rozbité tabuľky do jednej

S normalizáciou treba narábať veľmi opatrne, lebo zmena schémy znamená zmenu všelikde. „Zdravý“ počiatočný návrh je BCNF, vyrobená z 3NF, ktorá zachováva „čo najviac“ funkčných závislostí. Výnimkou z BCNF môžu byť pochybné závislosti typu PSC→Mesto. (Ak BCNF láme funkčné závislosti, je možno rozumné ostať pri 3NF, prípadne snažiť sa vyrobiť inú 3NF/BCNF dekompozíciu.)