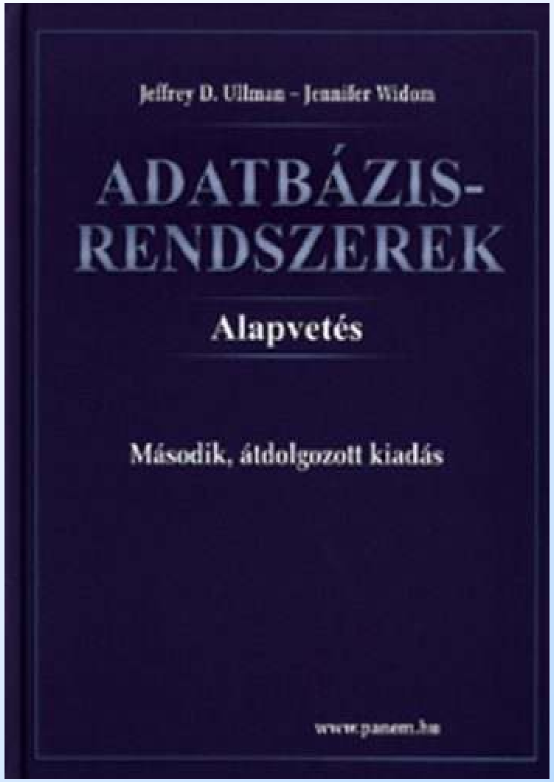
Relációs adatbázisok tervezése
2.rész (dekompozíció)



Ullman-Widom: Adatbázisrendszerek
Alapvetés. Második, átdolgozott kiadás,
Panem Kiadó, 2009

3.3. Relációs adatbázissémák tervezése
- Anomáliák, relációk felbontása

3.4. Dekompozíció tulajdonságai
- Veszteségmentes, függőségőrző

(Jeffrey D. Ullman 2007 előadásdiái alapján,
Benczúr András, Kiss Attila és Kósa Balázs
előadásainak felhasználásával, Hajas Csilla)

Relációs sémák tervezése

- Cél: az anomáliák és a redundancia megszüntetése.
  - *Módosítási anomália* : egy adat egy előfordulását megváltoztatjuk, más előfordulásait azonban nem.
  - *Törlési anomália* : törléskor olyan adatot is elveszítünk, amit nem szeretnénk.
  - *Beillesztési anomália* : megszorítás, trigger kell, hogy ellenőrizni tudjuk (pl. a kulcsfüggőséget)
  - *REDUNDANCIA* (többszörözés feleslegesen)

Példa: rosszul tervezett séma

Sörivó(<u>név</u>, cím, <u>kedveltSörök</u>, gyártó, kedvencSör)

név	cím	kedveltSörök	gyártó	kedvencS
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	???	WickedAle	Pete's	???
Spock	Enterprise	Bud	???	Bud

Redundáns adat, a ??? helyén a
név -> cím kedvencSör és kedveltSörök -> gyártó
FF-ek felhasználásával tudjuk, mi szerepel.

A rosszul tervezettség anomáliákat is eredményez

Sörivó(<u>név</u>, cím, <u>kedveltSörök</u>, gyártó, kedvencSör)

név	cím	kedveltSörök	gyártó	kedvencS
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	???	WickedAle	Pete's	???
Spock	Enterprise	Bud	???	Bud

- Módosítási anomália: ha Janeway-t *Karcsira* módoítjuk, megtesszük-e ezt minden sornál?
- Törlési anomális: Ha senki sem szereti a Bud sört, azt sem tudjuk, ki gyártotta.

Dekomponálás

A fenti problémáktól dekomponálással (felbontással) tudunk megszabadulni!

Definíció:

$d=\{R_1,...,R_k\}$ az (R,F) dekompozíciója, ha nem marad ki attribútum, azaz $R_1 \cup ... \cup R_k = R$. (Az adattábla felbontását projekcióval végezzük).

Például:

R=ABCDE, d={AD,BCE,ABE}

3 tagú dekompozíció, ahol

R_1=AD, R_2=BCE, R_3=ABE,

Dekomponálás

- Elvárások (Mikor használható?)

(1) Anomáliák kiküszöbölése. A vetületek legyenek egyszerűek, jó tulajdonságúak (normálformák: BCNF, 3NF)

(2) Információ-visszaállíthatóság, vagyis ne legyen információvesztés, veszteségmentes felbontások

(3) Függőségek megőrzése a vetületekben

Veszteségmentes szétvágás I.

- A fenti jelölésekkel: ha $r = \Pi_{R1}(r) \bowtie \ldots \bowtie \Pi_{Rk}(r)$ teljesül, akkor az előbbi összekapcsolásra azt mondjuk, hogy veszteségmentes. Itt r egy R sémájú relációt jelöl.
- Megj.: könnyen látható, hogy $r \subseteq \Pi_{R1}(r) \bowtie \ldots \bowtie \Pi_{Rk}(r)$ mindig teljesül. (Miért?)

R

A	B	C
a	b	c
d	e	f
c	b	c

R_1

A	B
a	b
d	e
c	b

R_2

B	C
b	c
e	f

Példa

- A szétvágás után keletkező relációk összekapcsolása nem veszteségmentes:

R

A	B	C
a	b	c
c	b	e

R_1

A	B
a	b
c	b

R_2

B	C
b	c
b	e

Chase-teszt veszteségmentességhez I.

- Példa: adott R(A, B, C, D), F = { $A \rightarrow B$, $B \rightarrow C$, $CD \rightarrow A$ } és az R_1(A, D), R_2(A, C), R_3(B, C, D) felbontás. Kérdés veszteségmentes-e a felbontás?
- Vegyük $R_1 \bowtie R_2 \bowtie R_3$ egy t = (a, b, c, d) sorát. Bizonyítani kell, hogy t R egy sora. A következő tablót készítjük:

A	B	C	D
a	b_1	c_1	d
a	b_2	c	d_2
a_3	b	c	d

Itt pl. az (a, b_1, c_1, d) sor azt jelzi, hogy R-nek van olyan sora, aminek R_1-re való levetítése (a, d), ám ennek a B és C attribútumokhoz tartozó értéke ismeretlen, így egyáltalán nem biztos, hogy a t sorról van szó.

Chase-teszt veszteségmentességhez II.

- Az F-beli függőségeket használva egyenlővé tesszük azokat a szimbólumokat, amelyeknek ugyanazoknak kell lennie, hogy valamelyik függőség ne sérüljön.
  - Ha a két egyenlővé teendő szimbólum közül az egyik index nélküli, akkor a másik is ezt az értéket kapja.
  - Két indexes szimbólum esetén a kisebbik indexű értéket kapja meg a másik.
  - A szimbólumok minden előfordulását helyettesíteni kell az új értékkel.
- Az algoritmus véget ér, ha valamelyik sor t-vel lesz egyenlő, vagy több szimbólumot már nem tudunk egyenlővé tenni.

Chase-teszt veszteségmentességhez III.

A	B	C	D
a	b_1	c_1	d
a	b_2	c	d_2
a_3	b	c	d

$A \rightarrow B$

A	B	C	D
a	b_1	c_1	d
a	b_1	c	d_2
a_3	b	c	d

$B \rightarrow C$

A	B	C	D
a	b_1	c	d
a	b_1	c	d_2
a_3	b	c	d

$CD \rightarrow A$

A	B	C	D
a	b_1	c	d
a	b_1	c	d_2
a	b	c	d

Chase-teszt veszteségmentességhez IV.

- Ha t szerepel a tablóban, akkor valóban R-nek egy sora, s mivel t-t tetszőlegesen választottuk, ezért a felbontás veszteségmentes.
- Ha nem kapjuk meg t-t, akkor viszont a felbontás nem veszteségmentes.
- Példa: R(A, B, C, D), F = { $B \rightarrow AD$ }, a felbontás: R_1(A, B), R_2(B, C), R_3(C, D).

A	B	C	D
a	b	c_1	d_1
a_2	b	c	d_2
a_3	b_3	c	d

$B \rightarrow AD$

A	B	C	D
a	b	c_1	d_1
a	b	c	d_1
a_3	b_3	c	d

Itt az eredmény jó ellenpélda, hiszen az összekapcsolásban szerepel t = (a, b, c, d), míg az eredeti relációban nem.

Függőségek megőrzése

- A dekompozíciókban érvényes függőségekből következzen az eredeti sémára kirótt összes függőség.
- Milyen függőségek lesznek érvényesek a dekompozíció sémáiban?
- Példa: R=ABC, F= $\{A\rightarrow B,\ B\rightarrow C\ ,\ C\rightarrow A\}$ vajon a d= (AB, BC) felbontás megőrzi-e a $C\rightarrow A$ függőséget? (függőségek vetülete, definíció a következő lapon)

Függőségek megőrzése

- Definíció: Függőségek vetülete:

  Adott (R,F), és $R_i \subseteq R$ esetén:

  $\Pi_{R_i}(F) := \{ X \rightarrow Y \mid F \vdash X \rightarrow Y , XY \subseteq R_i \}$

- Definíció: Adott (R,F) esetén $d=(R_1,\ldots,R_k)$ <u>függőségőrző dekompozíció</u> akkor és csak akkor, ha minden F-beli függőség levezethető a vetületi függőségekből:

  minden $X \rightarrow Y \in F$ esetén

  $$\Pi_{R_1}(F) \cup \ldots \cup \Pi_{R_k}(F) \vdash X \rightarrow Y$$

Példa: függőségek vetülete

- ABC, $A \text{->} B$ és $B \text{->} C$ FF-kel. Projektáljunk AC-re.
  - $A^+ = ABC$; ebből $A \text{->} B$, $A \text{->} C$.
    - Nem kell kiszámítani AB^+ és AC^+ lezárásokat.
  - $B^+ = BC$; ebből $B \text{->} C$.
  - $C^+ = C$; semmit nem ad.
  - $BC^+ = BC$; semmit nem ad.
- A kapott FF-ek: $A \text{->} B$, $A \text{->} C$ és $B \text{->} C$.
- AC-re projekció: $A \text{->} C$.

Függőségek megőrzése

- A függőségőrzésből nem következik a veszteségmentesség:

  R=ABCD, F= {A$\rightarrow$B,C$\rightarrow$D}, d={AB,CD} függőségőrző, de nem veszteségmentes.

- A veszteségmentességből nem következik a függőségőrzés:

  R=ABC, F= {AB$\rightarrow$C,C$\rightarrow$A}, d={AC,BC} veszteségmentes, de nem függőségőrző.

Függőségek kikényszerítése

- Bizonyos FF halmazok esetén a felbontáskor elveszíthetünk függőségeket. Legyen ABC.
- *AB* -> *C* és *C* -> *B*.
  - Példa: *A* = utca, *B* = város, *C* = irányítószám.
- Két kulcs van: {*A*,*B* } és {*A*,*C* } /nem BCNF/
- A probléma az, hogy *AC* és *BC* sémákkal és semmilyen más felbontással nem tudjuk kikényszeríteni az *AB* -> *C* függőséget.

Egy kikényszeríthetetlen FF

utca	iSzám
545 Tech Sq.	02138
545 Tech Sq.	02139

város	iSzám
Cambridge	02138
Cambridge	02139

Kapcsoljuk össze a sorokat (iSzám).

street	city	zip
545 Tech Sq.	Cambridge	02138
545 Tech Sq.	Cambridge	02139

A szétbontott relációkban egyik FF sem sérül, az eredményben az utca város -> iSzám nem teljesül.