Adatbázisok 1. Relációs adatbázis tervezés – 4. rész

Funkcionális függőségek

Felbontások

Normálformák

Relációs sémák tervezése

- Cél: az anomáliák és a redundancia megszüntetése.
  - *Módosítási anomália*
  - *Törlési anomália*
  - *Beszúrási anomália*

Példa: rosszul tervezett séma

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

név	cím	kedveltSörök	gyártó	kedvencSör
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

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

- Módosítási anomália: ha Janeway-t *Judyra* módosítjuk, megtesszük-e ezt minden sornál?
- Törlési anomália: ha már senki sem szereti a Bud sört, azt sem fogjuk tudni, hogy ki gyártotta.
- Beszúrási anomália: ha nem lehet ismeretlen a kedveltSörökre vonatkozó érték, akkor nem tudunk felvenni egy antialkoholista főnököt

Relációk felbontása

- R relációt, $\{A_1, A_2, \ldots, A_n\}$ attribútumokkal helyettesítsük $S(B_1, B_2, \ldots, B_m)$ és $T(C_1, C_2, \ldots, C_k)$ relációkkal úgy, hogy

1. $\{A_1, A_2, \ldots, A_n\} = \{B_1, B_2, \ldots, B_m\} \cup \{C_1, C_2, \ldots, C_k\}$
2. $S = \Pi_{B1,\ B2,\ \ldots,\ Bm}(R)$
3. $T = \Pi_{C1,\ C2,\ \ldots,\ Ck}\ (R)$

Veszteségmentes felbontás

- Ha $r = \Pi_{R1}(r)$ |X| … |X| $\Pi_{Rk}(r)$ teljesül, akkor az előbbi összekapcsolásra azt mondjuk, hogy veszteségmentes. Itt r egy R sémájú relációt jelöl.
- $\Pi_{Ri}(r)$ jelentése: r sorai az Ri attribútumaira projektálva.
- Megjegyzés: könnyen látható, hogy $r \subseteq \Pi_{R1}(r)$ |X| … |X| $\Pi_{Rk}(r)$ 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

Boyce-Codd normálforma

- *R* reláció *BCNF* normálformában van, ha minden *X ->Y* nemtriviális FF-re *R*-ben *X* szuperkulcs.
  - *Nemtriviális:* *Y* nem része *X*-nek.
  - *Szuperkulcs:* tartalmaz kulcsot (ő maga is lehet kulcs).

Példa

Főnökök(név, cím, kedveltSörök, gyártó, kedvencSör)

FF-ek: név->cím kedvencSör, kedveltSörök->gyártó

- Itt egy kulcs van: {név, kedveltSörök}.
- A baloldalak egyik FF esetén sem szuperkulcsok.
- Emiatt az *Főnökök* reláció nincs BCNF normálformában.

Még egy példa

Sörök(<u>név</u>, gyártó, gyártóCím)

FF-ek: név->gyártó, gyártó->gyártóCím

- Az egyetlen kulcs {név} .
- név->gyártó nem sérti a BCNF feltételét, de a gyártó->gyártóCím függőség igen.

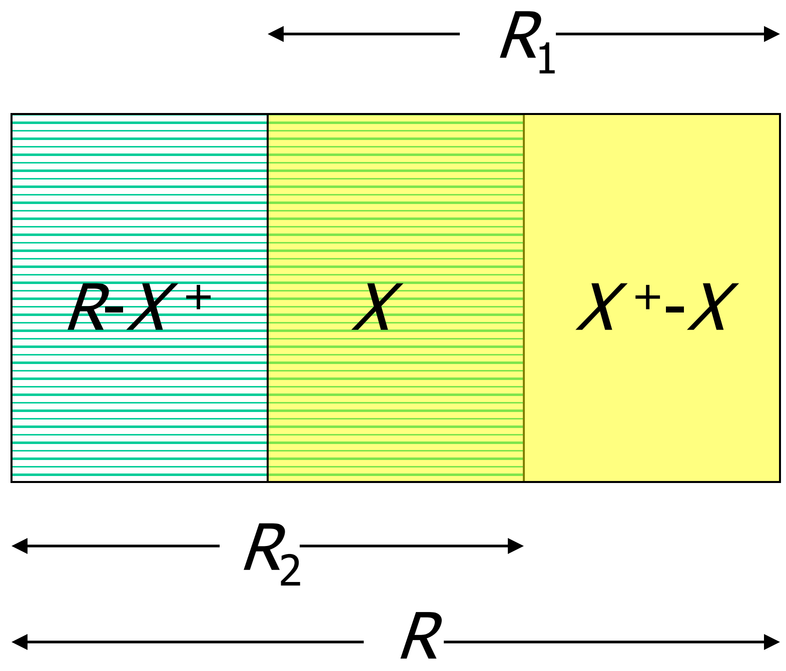
BCNF-re való felbontás

- Adott *R* reláció és *F* funkcionális függőségek.
- Van-e olyan $X \rightarrow Y$ FF, ami sérti a BCNF-t?
  - Ha van olyan következmény FF *F*-ben, ami sérti a BCNF-t, akkor egy *F*-beli FF is sérti.
- Kiszámítjuk X^+-t:
  - Ha itt nem szerepel az összes attribútum, *X* nem szuperkulcs.

R dekomponálása X -> Y felhasználásával

- R-t helyettesítsük az alábbiakkal:
  1. $R_1 = X^+$.
  2. $R_2 = R - (X^+ - X)$.
- *Projektáljuk* a meglévő F -beli FF-eket a két új relációsémára.

Dekomponálási kép



Példa: BCNF dekompozíció

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

F = név->cím, név->kedvencSör,

kedveltSörök->gyártó

- Vegyük név->cím FF-t:
- $\{név\}^+ = \{név, cím, kedvencSör\}$.
- A dekomponált relációsémák:
  1. Főnökök1(<u>név</u>, cím, kedvencSör)
  2. Főnökök2(<u>név</u>, <u>kedveltSörök</u>, gyártó)

Példa -- folytatás

- Meg kell néznünk, hogy az Főnökök1 és Főnökök2 táblák BCNF-ben vannak-e.
- Az FF-ek projektálása könnyű.
- A Főnökök1(név, cím, kedvencSör), az FF-ek név->cím és név->kedvencSör.
  - Tehát az egyetlen kulcs: {név}, azaz az Főnökök1 BCNF-ben van.

Példa -- folytatás

- Az Főnökök2(név, kedveltSörök, gyártó) esetén az egyetlen FF

  kedveltSörök->gyártó, az egyetlen kulcs: {név, kedveltSörök}.

  - Sérül a BCNF.

- kedveltSörök$^+$ = {kedveltSörök, gyártó}, az *Főnökök2* felbontása:

1. Főnökök3(kedveltSörök, gyártó)
2. Főnökök4(név, kedveltSörök)

Példa -- befejezés

- Az *Főnökök* dekompozíciója tehát:
  1. Főnökök1(<u>név</u>, cím, kedvencSör)
  2. Főnökök3(<u>kedveltSörök</u>, gyártó)
  3. Főnökök4(<u>név</u>, <u>kedveltSörök</u>)
- Az *Főnökök1* az főnökökről, az *Főnökök3* a sörökről, az *Főnökök4* az főnökökről és kedvelt söreikről tartalmaz információt.

Példa – anomáliák?

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

1. Főnökök1(<u>név</u>, cím, kedvencSör)
2. Főnökök3(<u>kedveltSörök</u>, gyártó)
3. Főnökök4(<u>név</u>, <u>kedveltSörök</u>)

név	cím	kedvencSör
Janeway	Voyager	WickedAle
Spock	Enterprise	Bud

kedveltSörök	gyártó
Bud	A.B.
WickedAle	Pete's

név	kedveltSörök
Janeway	Bud
Janeway	WickedAle
Spock	Bud

Miért működik a BCNF?

- (R, F) esetén ha $R_1, \ldots, R_k$ egy veszteségmentes felbontás, S_1, S_2 pedig R_1 veszteségmentes felbontása, akkor S_1, S_2, $R_2, \ldots, R_k$ is veszteségmentes felbontás.
- Könnyen ellenőrizhető, hogy a BCNF felbontásos dián az R_1, R_2 veszteségmentes. Ehhez:

  Feladat: bizonyítsuk be, hogy ha az R(A, B, C) reláció esetén $B \rightarrow C$ teljesül, akkor az $R_1(A, B)$, $R_2(B, C)$ felbontás mindig veszteségmentes.
- Minden két attribútumú séma BCNF normálformában van.
- A fentiekkel igazolható:
- Az algoritmus tehát valóban veszteségmentes felbontást ad, ám sajnos exponenciális lépésszámú is lehet a függőségek vetítése miatt.

Chase-teszt veszteségmentességhez I.

- Példa: adott R(A, B, C, D), F = { A → B, B → C, CD → 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 |X| R_2 |X| R_3 egy *t* = (a, b, c, d) sorát. Bizonyítani kell, hogy *t* R egy sora. A következő tablót készítjük el:

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, és 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 → 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 → 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.

Chase-teszt veszteségmentességhez V.

- *{A ,B }*, {(a, b), (a_3, b_3)}.
- {*B ,C* }, {(b, c_1), (b, c), (b_3, c)},
- *{C, D}*, (c_1,d_1), (c, d_1), (c, d*)}*
- ⋈, {(a, b, c_1), (a, b, c), (a_3, b_3, c)}.
- ⋈, {(a, b, c_1,d_1), (a, b, c, d_1), (a, b, c, d), (a_3, b_3, c, d_1) , (a_3, b_3, c, d)}.}
- 2 extra sor (rekord), és (a, b, c, d), amit kell is tartalmaznia.