Adatbázisok elmélete 16. előadás

Katona Gyula Y.
Budapesti Műszaki és Gazdaságtudományi Egyetem
Számítástudományi Tsz.
I. B. 137/b
kiskat@cs.bme.hu
http://www.cs.bme.hu/~kiskat

2005

Többértékű függés (emlékeztető)

Definíció. *Az X attribútumhalmaztól* ***többértékűen függ*** *az Y attribútumhalmaz az r relációban (jele: $X \twoheadrightarrow Y$), ha tetszőleges $t_1, t_2 \in r$ sorokra, melyekre $t_1[X] = t_2[X]$, létezik $t_3, t_4 \in r$, melyekre*

- $t_3[XY] = t_1[XY]$
- $t_3[R \setminus XY] = t_2[R \setminus XY]$
- $t_4[XY] = t_2[XY]$
- $t_4[R \setminus XY] = t_1[R \setminus XY]$

	X	Y	$R \setminus XY$
	....	....	....
t_1	AAAAAAA	BBBBBBB	CCCCCCC
t_2	AAAAAAA	DDDDDDD	EEEEEEE
	⋮	⋮	⋮
t_3	AAAAAAA	BBBBBBB	EEEEEEE
t_4	AAAAAAA	DDDDDDD	CCCCCCC

4NF (emlékeztető)

Tétel. *Legyen $\rho = (R_1, R_2)$ az (R, F) séma felbontása, ahol F most funkcionális és többértékű függéseket is tartalmaz. ρ akkor és csak akkor hűséges felbontás, ha $R_1 \cap R_2 \twoheadrightarrow R_2 \setminus R_1$.*

Definíció. *Az (R, F) séma* ***4NF*** *(****negyedik normálformájú****), ha tetszőleges nemtriviális $X \twoheadrightarrow Y \in F^+$ esetén X szuperkulcs (a szuperkulcsot a régi értelemben, csak funkcionális függőségekkel definiálva).*

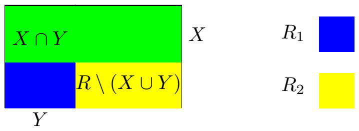
Felbontás 4NF-re

Tétel. *Legyen (R, F) egy séma, ahol F funkcionális és többértékű függések halmaza. Ekkor (R, F) felbontható hűségesen 4NF relációkra.*

Algoritmus: Hasonlóan BCNF-hez, mindig két valódi részre bontjuk hűségesen, addig, amíg mindegyik rész 4NF nem lesz.

Keresünk egy $X \twoheadrightarrow Y$ függést, ami megsérti a 4NF feltételt.
(Ha van $\rightarrow$, ami megsérti, akkor van $\twoheadrightarrow$ is.)
Nem tanuljuk, hogy ezt hogy kell általában, mert bonyolult, de ha nem kell keresni, mert ott van, akkor meg tudjuk csinálni (ezt tudni kell majd ZH-n, vizsgán)

$$R_1 = XY \qquad R_2 = R \setminus (Y \setminus X) \ (= X \cup (R \setminus Y))$$



Ez valódi felbontás:
Ha $R_1 = R \implies X \twoheadrightarrow Y$ triviális függés lenne, ↯.
Ha $R_2 = R \implies Y \subseteq X \implies X \twoheadrightarrow Y$ triviális függés lenne, ↯.
Ez hűséges felbontás:
$R_1 \cap R_2 = X$; $R_2 \setminus R_1 = R \setminus XY$ és $X \twoheadrightarrow R \setminus XY$ fennáll $X \twoheadrightarrow Y$ miatt.

Példa

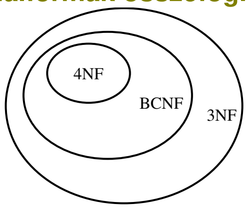
R(Színész, Város, Utca, Filmcím, Filmév)

F = {Színész $\twoheadrightarrow$ Város, Utca}

Ez megsérti a 4NF tulajdonságot, ha Színész nem szuperkulcs.

4NF felbontás: R_1 =(Színész, Város, Utca) R_2 =(Színész, Filmcím, Filmév)

Normálformák összefoglalása



Jellemzők	3NF	BCNF	4NF
Megszünteti a funkcionális függőségekből eredő redundanciát	Gyakran	Igen	Igen
Megszünteti a többértékű függőségekből eredő redundanciát	Nem	Nem	Igen
Az ilyen felbontás megőrzi a funkcionális függőségeket	Igen	Lehet	Lehet
Az ilyen felbontás megőrzi a többértékű függőségeket	Lehet	Lehet	Lehet

Fontos elv: Ne bontsuk tovább, amit már nem muszáj.
A normalizálás azért fontos, mert ...

Eddig tart a ZH anyaga!

Adatbázisrendszerek megvalósítása

Eddig az adatbáziskezelők működéséről tanultunk. Az év hátralevő részében az ilyen rendszerek belső működését tanulmányozzuk egy kicsit.

Három nagyobb témakör:

1. Lekérdezésfeldolgozás: hogyan értékelődnek ki a lekérdezések, milyen módszerek vannak a lekérdezések végrehajtására?
2. Fizikai szervezés, tárkezelés: hogyan tároljuk a relációkat oly módon, hogy gyorsan lehessen keresni, illetve módosítani?
3. Tranzakciókezelés: többfelhasználós működés biztosítása, illetve rendszerhibák elleni védelem.

Lekérdezések végrehajtása, „optimalizálása”

Elemzés (parsing):

- szintaktikai ellenőrzés $\Longrightarrow$ *megfelelő parancsok, megfelelő sorrendben*
- átírás elemzőfa alakra

Előfeldolgozó:

- Relációk használatának ellenőrzése $\Longrightarrow$ *van-e ilyen*
- Attribútumnevek használatának ellenőrzése $\Longrightarrow$ *pl. egyértelmű-e, melyik attribútum melyik relációban van, benne van-e egyáltalán*
- típusellenőrzések $\Longrightarrow$ *pl. LIKE használatakor csak karakterlánc lehet*

Logikai lekérdezési terv:

- Átírás (kibővített) relációs algebrai alakra
- Transzformációk $\Longrightarrow$ *több terv, gyorsítás*
- Legjobb terv kiválasztása költségbecsléssel

Fizikai terv kiválasztása:

- Algoritmusok a műveletekhez
- Pufferkezelés
- Közbülső relációk eltárolása

A relációs algebra kibővítése

Az SQL többet tud, mint a relációs algebra, de az extra dolgokat is át akarjuk írni relációs formába. Néhány különbség:

- Multihalmazok $\Longrightarrow \cap_H, \cap_M$
- Kiválasztásnál, $\bowtie_\theta$-nál a feltételben használhatunk aritmetikai műveleteket $\Longrightarrow \sigma_{A+B<5}(R),\ R \underset{A+R.B<C+S.B}{\bowtie} S$
- Vetítés aritmetikai műveletekkel és átnevezéssel $\Longrightarrow \pi_{A,B+C\to X}(R)$
- Ismétlődések kiszűrése $\Longrightarrow \delta(R)$
- Csoportosítások, aggregátumok $\Longrightarrow$ SELECT $A, MIN(B)$ AS $minB$ FROM R GROUP BY A $\Longrightarrow \gamma_{A,MIN(B)\to minB}(R)$

Fizikai végrehajtás

Leginkább az I/O műveletigény érdekes. Ha „túl nagy” a számítási igény az is baj lehet.

Soronkénti, unáris műveletek: Kiválasztás és vetítés. Egyszerre csak egy sort kell vizsgálni, az algoritmus nem függ a memória nagyságától.

Unáris, teljes relációs műveletek: Pl. $\delta(R), \gamma(R)$. Ha nem fér el a reláció a memóriában, akkor mást kell csinálni.

Bináris, teljes relációs műveletek: $\cup, \cap, \setminus, \times, \bowtie$. Sok minden függ a méretektől.

Fizikai végrehajtás

$R(X,Y) \bowtie S(Y,Z)$ végrehajtása:

- *Ha $B(S) < M-1$, azaz S belefér a memóriába:* egymenetes algoritmus
  1. Beolvassuk S-et és hashtáblát vagy B-fát készítünk, ahol a kulcs Y attribútumai.
  2. Beolvasunk egy blokkot R-ből. Minden sorára kikeressük a passzoló S-beli sorokat. Az eredményt kiírjuk.

  I/O műveletigény: $B(S) + B(R)$
- *Ha $B(R) > B(S) > M-1$:* beágyazott ciklusú algoritmus
  Beolvasunk minél több blokkot a memóriába S-ből, utána ugyanazt csináljuk mint fenn.
  I/O műveletigény: $B(S) + B(S)B(R)/(M-1) \approx B(S)B(R)/M$
- *Ha $B(R), B(S) \leq M^2$:* rendezéses algoritmus
  Y kulcs szerint rendezzük R-et és S-et összefésüléses rendezéssel. Vesszük az összes y kulcsú sort a két lista elejéről és kiírjuk az összes párt. (Feltettük, hogy az összes y kulcsú sor elfér a memóriában.)
  I/O műveletigény: $5(B(S) + B(R))$

Fizikai végrehajtás

- *Ha $\min(B(R), B(S)) \leq M^2$:* hasheléses algoritmus
  Y kulcs szerint vödrös hashelést végzünk R-re és S-re. (Ha közben megtelik egy vödör, azt kiírjuk.) A kapott R_i, S_i vödrökkel egymenetes algoritmust végzünk.
  I/O műveletigény: $3(B(S) + B(R))$
- *Ha van index S-re Y szerint:* indexet használó algoritmus
  R-et blokkonként olvassuk be, az index alapján keressük ki a hozzá passzoló sorokat.
  Átlagos I/O műveletigény: $B(S)B(R)/V(S,Y)$, ahol $V(S,Y)$: Y értékkészletének számossága S-ben.