Adatbázisok elmélete 8. 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`

2004

Példák oszlopkalkulus alkalmazására

ÁRU(ÁRUKÓD, ÁRUNÉV, EGYSÉGÁR)
MENNYISÉG(DÁTUM, ÁRUKÓD, DB)
BEVÉTEL(DÁTUM, ÖSSZEG)
BEFIZ(ÖSSZEG, BEFIZ) BEFIZ=ÖSSZEG−**4000**

Hány darabot adtak el 2004. jan. 15-én az A123 kódú áruból, mi a neve és az ára?

$$\left\{s^{(3)} \mid \exists u \exists v \Big(\text{MENNYISÉG}(u) \wedge \text{ÁRU}(v) \wedge u[1] = \text{2004-01-15} \wedge u[2] = \text{'A123'} \wedge v[1] = \text{'A123'} \wedge s[1] = u[3] \wedge s[2] = v[2] \wedge s[3] = v[3]\Big)\right\}$$

$$\left\{x, y, z \mid \exists u \exists v \Big(\text{MENNYISÉG}(u, v, x) \wedge \text{ÁRU}(v, y, z) \wedge u = \text{2004-01-15} \wedge v = \text{'A123'}\Big)\right\}$$

Mely nevű áruk azok, amelyekkel van azonos egységárú másik áru?

$$\left\{s^{(1)} \mid \exists u \exists v \Big(\text{ÁRU}(v) \wedge \text{ÁRU}(u) \wedge s[1] = v[2] \wedge v[3] = u[3] \wedge \neg(v[1] = u[1])\Big)\right\}$$

$$\left\{v \mid \exists x \exists y \exists w \exists u \Big(\text{ÁRU}(x, v, u) \wedge \text{ÁRU}(y, w, u) \wedge x \neq y\Big)\right\}$$

Biztonságos kifejezés

Dom(ψ) és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.
Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Tétel. *A sorkalkulus és az oszlopkalkulus ekvivalensek. A biztonságos sorkalkulus és a biztonságos oszlopkalkulus ekvivalensek.*

Bizonyítás: *Vázlat:*

$$\begin{array}{rcl}
\left\{t^{(k)} \mid \phi\left(t^{(k)}\right)\right\} & \longleftrightarrow & \{u_1, \ldots, u_k \mid \psi(u_1, \ldots, u_k)\} \\
t^{(k)} & \longleftrightarrow & u_1, \ldots, u_k \\
R\left(t^{(k)}\right) & \longleftrightarrow & R(u_1, \ldots, u_k) \\
t^{(k)}[j] & \longleftrightarrow & u_j \\
\exists t^{(k)} \phi\left(t^{(k)}\right) & \longleftrightarrow & \exists u_1 \ldots \exists u_k \psi(u_1, \ldots, u_k) \\
\text{biztonságos} & \longleftrightarrow & \text{biztonságos}
\end{array}$$

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek algebrai alapúak: relációs algebrán alapuló lekérdezés, procedurális leírás
  Előny: lekérdezésoptimalizálásra jobb
- Lehetnek logikai alapúak: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
  Előny: könnyebben átlátható

Általában a konkrét lekérdezőnyelvek eltérnek a modelltől (algebrai és logikai esetben is), van amiben többet tudnak, van amiben kevesebbet, vagy csak máshogy.

Lehetséges eltérések:

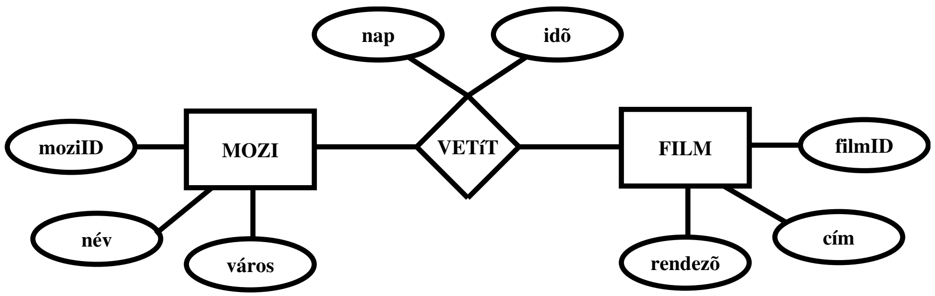
- logikai alapúakban eleve csak biztonságos kifejezéseknek megfelelő kérdéseket lehet írni (nincsenek kvantorok)
- aritmetika
- aggregátumok
- többszörös sorok kezelése/megengedése
- attribútumok sorrendje kötött

De az igaz mindre, hogy relációsan teljesek (esetleg bizonyos műveletek nehezebben mennek) és általában van a lekérdező funkció mellett DML és DDL is.

Példák relációs adatbáziskezelő nyelvekre

- Information System Base Language $\Longrightarrow$ ISBL
  - ★ relációs algebra alapú
  - ★ kifejlesztő: IBM's United Kingdom Scientific Center
  - ★ Peterlee Relational Test Vehicle
- QUEry Language $\Longrightarrow$ QUEL
  - ★ sorkalkulus alapú
  - ★ kifejlesztő: University of California at Berkly
  - ★ INGRES lekérdezőnyelve
- Query-By-Example $\Longrightarrow$ QBE
  - ★ oszlopkalkulus alapú
  - ★ kifejlesztő: IBM's Watson Research Center
  - ★ pl. MS Access
- Structured Query Language $\Longrightarrow$ SQL
  - ★ oszlopkalkulus-szerű alapjai vannak, némi sorkalkulus elemmel
  - ★ kifejlesztő: IBM's San Jose Research Laboratory
  - ★ pl. Oracle, MS SQL, IBM DB2, MySQL, PostgreSQL

A példákban használt relációs séma



MOZI	moziID	név	város
	101	Művész	Budapest
	102	Uránia	Pécs
	⋮		

FILM	filmID	cím	rendező
	1	Macskajaj	E. Kusturica
	2	Moszkva tér	Török F.
	⋮		

VETÍT	moziID	filmID	nap	idő
	101	1	péntek	16:00
	101	2	szombat	19:00
	⋮			

ISBL

- Relációs algebra alapú lekérdezések
  - ★ operátorok

művelet	relációs algebra	ISBL
unió	$R \cup S$	**R + S**
metszet	$R \cap S$	**R . S**
természetes illesztés	$R \bowtie S$	**R * S**
különbség (általánosabb)	$R \setminus (R \ltimes S)$	**R – S**
kiválasztás	$\sigma_F(R)$	**R:F**
vetítés	$\pi_{A,B,C}(R)$	**R % A,B,C**

  - ★ egyéb elemek
    - eredmény megjelenítése: **list** kulcsszó
    - értékadás relációnak: **=**
    - és, vagy, tagadás jelei: **&, |, ¬**
- További nyelvi elemek
  - ★ aggregátumok (min., max., összeg, átlag, darabszám) kezelése
  - ★ adatfrissítő műveletek
  - ★ átnevezés
  - ★ kimenet formázása

ISBL példák

- Budapesti mozik nevei
  list MOZI **:** város = "Budapest" **%** név
- Pénteken 16 órakor kezdődő filmek címe, rendezője
  list FILM * VETÍT **:** nap = "péntek" & idő = "16:00" **%** cím, rendező
- Pénteken nem vetített filmek címe
  list FILM **–** (VETÍT **:** nap = "péntek") **%** cím
  (Figyelem! Különbségnek más a definíciója.)
- Pénteken és szombaton is vetített filmek címe
  list (FILM * VETÍT **:** nap = "péntek" **%** cím) .
  (FILM * VETÍT **:** nap = "szombat" **%** cím)
  vagy köztes relációk bevezetésével:
  r1 **=** FILM * VETÍT **:** nap = "péntek" **%** cím
  r2 **=** FILM * VETÍT **:** nap = "szombat" **%** cím
  list r1.r2

QUEL

Sorkalkulus alapú.

Pl. Budapesti mozik nevei:

$\{t[2] \mid \mathrm{MOZI}(t) \wedge t[3] = \text{'Budapest'}\}$

Meg kell mondani, hogy a sorváltozó melyik reláció sorain fut:

$$R(t) \quad \longleftrightarrow \quad \textbf{range of } t \textbf{ is } R$$

Hivatkozni kell a sorváltozó komponenseire:

$$t[i] \quad \longleftrightarrow \quad \textbf{t.<}i\textbf{-edik attributum neve>}$$

Pl. $t[3] \longleftrightarrow t$.város

Lekérdezés:

retrieve (<lekérdezendő attribútumok> **) where** <feltétel>

QUEL példák

range of m **is MOZI**
range of f **is FILM**
range of v **is VETÍT**
range of u **is VETÍT**

- Budapesti mozik nevei
  retrieve (m.név **) where** m.város ='Budapest'
- Pénteken 16 órakor kezdődő filmek címe, rendezője
  retrieve unique (f.cím, f.rendező **)**
  where f.filmID=v.filmID **and** v.nap = 'péntek' **and** v.idő = '16:00'
- Pénteken nem vetített filmek címe:
  $\left\{t^{(3)}[2] \mid \mathrm{FILM}(t) \wedge \nexists s\left(\mathrm{VETÍT}(s) \wedge s[2] = t[1] \wedge s[3] = \text{'péntek'}\right)\right\}$
  QUEL-ben nincsenek igazi kvantorok, csak egy **any** nevű aggregatum!
  retrieve (f.cím **)**
  where any (f.filmID **where** f.filmID=v.filmID **and** v.nap = 'péntek'**)**=0

Lehet beszúrni és törölni:
retrieve into FILM1 **unique** f.cím
retrieve into FILM2 **unique** f.cím **where** f.filmID=v.filmID **and** v.nap='péntek'
range of $f1$ **is** FILM1
range of $f2$ **is** FILM2
delete $f1$ **where** $f1$.filmID=$f2$.filmID
retrieve FILM1

- Pénteken és szombaton is vetített filmek címe
  retrieve unique (f.cím **)**
  where f.filmID=v.filmID **and** $v1$.filmID=v.filmID **and**
  v.nap = 'péntek' **and** $v1$.nap = 'szombat'

QBE

- Oszlopkalkulus alapú lekérdezések, kétdimenziós
  - ★ A lekérdezés elemei változókkal és konstansokkal kitöltött sablon(ok)
    - ⁎ jelölések

változó	aláhúzott változónév
konstans	nem aláhúzott érték
egyszer említett változó	üres cella
kimenetre kerülő attribútum	P. prefix

    - ⁎ Példa: Budapesti mozik nevei:

MOZI	moziID	név	város
	$\underline{1}$	P.$\underline{\text{mozinév}}$	Budapest

vagy

MOZI	moziID	név	város
		P.	Budapest

  - ★ Összetett lekérdezések is lehetségesek (használatukkor az azonos nevű változók illesztése történik meg).
    - ⁎ használható több soros sablon (ekkor a kiértékeléskor mindegyik sornak egy-egy futó oszlopváltozó fog megfelelni, és ha illeszkedés van, akkor megtörténik a kiírás)
    - ⁎ használható több sablon (kiértékelés hasonlóan, mint a többsoros kérdésnél, csak az oszlopváltozók nem ugyanazon reláció sorait futják be)

- ★ A kiválasztás feltételeinek megadása
  - ⁎ Az egyenlőség konstanshoz való illesztéssel vizsgálható (mint az előbb),
  - ⁎ egyéb egyszerű relációkhoz a ¬ =, >, <, >=, <, <= operátorok használhatók,
  - ⁎ összetett feltételeket (pl. két változó közt a < relációt) külön feltételsablon megadásával lehet vizsgálni.
- További nyelvi elemek
  - ★ mintaillesztés
  - ★ aritmetika
  - ★ kimenet rendezése
  - ★ csoportosítás
  - ★ aggregátumok kezelése
  - ★ reláció tranzitív lezártjának kezelése
  - ★ adatmódosító műveletek
  - ★ típusdefiníció, sémalétrehozás

Megjegyzés: dupla példányt kiirtja, azaz többszörös sorok nincsenek

QBE példák még

- Nem budapesti mozik nevei

MOZI	moziID	név	város
		P.	¬ = Budapest

- Pénteki és szombati kezdési időpontok

VETÍT	moziID	filmID	nap	idő
			P.péntek	P.
			P.szombat	P.

- Időpontok, amikor pénteken és szombaton is kezdődik film

VETÍT	moziID	filmID	nap	idő
			péntek	P.$\underline{\text{kezdes}}$
			szombat	$\underline{\text{kezdes}}$

- Pénteken vetített filmek adatai

FILM	filmID	cím	rendező
	$\underline{1}$	P.	P.

VETÍT	moziID	filmID	nap	idő
		$\underline{1}$	péntek	

- Azok a városok, ahol van legalább két mozi:

MOZI	moziID	név	város
	$\underline{1}$		P.$\underline{\text{városnév}}$
	$\underline{2}$		$\underline{\text{városnév}}$

CONDITIONS
$\underline{1}\neq\underline{2}$

Az SQL nyelv

- Relációs nyelv, mint az eddigiek
- oszlopkalkulus jellegű, de némi sorkalkulusos beütéssel

Termékek, (amik szükségszerűen relációs nyelvet is tartalmaztak):

- IBM: System/R
- Relational Software: Oracle
- Relational Systems: Ingres

Szabványok

- SQL89 (SQL1)
- SQL92 (SQL2, mi nagyrészt ezt nézzük most)
- SQL99 (SQL3, ebből is pár dolog, pl. triggerek, rekurzió)

Működő rendszerekben ezek verziói vannak (főleg SQL2).

Fontosabb utasítások

Data Definition Language:

- CREATE - séma létrehozása
- ALTER - séma módosítása
- DROP - séma törlése

Data Modification Language:

- INSERT - adatok beszúrása
- UPDATE - adatok módosítása
- DELETE - adatok törlése
- SELECT - adatok lekérdezése

Természetesen előbb mindig a sémát kell létrehozni, és utána dolgozhatunk vele, de most fordítva tárgyaljuk mert eddig a lekérdező nyelvekről volt szó.

DML utasítások — SELECT

Ezzel valósítható meg a kiválasztás, vetítés és a szorzat.

Szintaxis: SELECT <reláció$_i$>.<attrib$_1$>, . . . , <reláció$_j$>.<attrib$_n$>
FROM <reláció$_1$>, . . . , <reláció$_m$>
WHERE <kifejezés>

Relációs algebrabeli megfelelője (de nem pontosan, mert SQL-ben SELECT nem küszöböli ki a többszörös sorokat):

$$\pi_{<attrib_1>,\dots,<attrib_n>}\sigma_{<kifejezés>}(<reláció_1> \times \cdots \times <reláció_m>)$$

Példa 1: A budapesti mozik azonosítói és nevei
SELECT mozi.moziID, mozi.név FROM mozi WHERE mozi.város="Budapest"

Példa 2: A pénteken hétkor kezdődő filmek azonosítói
SELECT vetít.filmID FROM vetít WHERE vetít.nap="péntek" AND vetít.idő="19:00"

Megjegyzés:

- kiértékelés: minden egyes FROM utáni relációnak megfelel egy-egy sorváltozó, ami az egyes relációk sorain megy végig (egymásba ágyazott ciklusokkal például). Ha találat van, azaz a WHEREfeltétel igaz az aktuális értékekre, akkor a SELECT utáni mezők kiíródnak
- úgy gondolhatunk a kiértékelésre, mintha először vennénk a FROM utáni relációk direkt szorzatát és aztan arra csinálnánk a kiválasztást és a vetítést.
- ha többszörös sorokat nem akarunk: SELECT DISTINCT (ennek ára van!!!)
- WHERE el is hagyható
- WHERE-ben mi állhat: erről később
- az eredmény az ORDER BY kulcsszó segítségével rendezhető, megadható hogy mely oszlopok szerint és hogy növőleg vagy csökkenőleg
- A fenti két példa mutatja, hogy a kiválasztás és a vetítés megy, a szorzatra a sorváltozók bevezetése után nézünk példát