Adatbázisok

Kormány Zsolt

2017. január 9.

A jegyzet és annak forrása megtalálható a `bme-notes.github.io` weboldalon.

Közreműködtek: Bognár Márton

Kellemes vizsgázást!

Tartalomjegyzék

Előszó

Lórem ipszum

Tételek es bizonyítások

Tétel: MZ/X használhatatlan cuccokat küld

Minden az $R_k^{(n_k)} \subseteq A^{n_k}, (k = 1, 2, \dots r)$ relációalgebrai kifejezéshez van olyan ψ sorkalkulus formula, hogy ψ csak az $R_k^{(n)}$-k közül tartalmaz relációkat és az E kifejezés megegyezik $\left\{s^{(m)} | \psi(s^{(m)})\right\}$

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

Az E kifejezésben található műveletek száma szerinti teljes indukvióval. n = 0 azaz nincs művelet E-ben, e csak egyetlen relációt tartalmazhat pl. $R_k^{(n_k)}$-t így

$E = \{s^{(n_k)} | R_k^{(n_k)}(s^{(n_k)}\}$ T.f.h E-ben **n** művelet van és az állítás még igaz. Igaz-e n+1 műveletre is?

Igaz volt tehát: $E_1 = \{t_1^{(n)} | \psi_1(t_1^{(n)})\}$ és $E_2 = \{t_2^{(m)} | \psi_2(t_2^{(m)})\}$

1. $\cup \to E = E_1 \cup E_2 (n = m) : E := \{t^{(n)} | \psi_1(t^{(n)}) \vee \psi_2(t_2^{(n)})\}$
2. $- \to E = E_1 - E_2 (n = m)$: $E := \{t^{(n)} | \psi_1(t^{(n)}) \wedge \neg\psi_2(t_2^{(n)})\}$
3. $\times \to E = E_1 \times E_2 (k = n + m) : E := \{t^{(k)} | \exists t_1^{(n)} \exists t_2^{(m)} \psi_1(t^{(n)}) \wedge \psi_2(t_2^{(n)}) \wedge t^{(k)}[1] = t_1^{(n)}[1] \wedge t^{(k)}[2] = t_1^{(n)}[2] \wedge \dots \wedge t^{(k)}[n] = t_1^{(n)}[n] \wedge t^{(k)}[n+1] = t_1^{(m)}[1] \wedge \dots \wedge t^{(k)}[n+m] = t_1^{(m)}[m]\}$
4. π $E = \pi_{i_1, i_2, \dots, i_r}(E_1) \to E := \{t^{(r)} | \exists u^{(n)} \psi_1(u^{(n)}) \wedge t[1] = u[i_1] \wedge t[2] = u[i_2] \wedge \dots \wedge t[r] = u[i_r]\}$
5. σ $E = \sigma_F(E_1) \to E := \{t^{(n)} | \psi_1(t^{(n)}) \wedge F'\}$ $F' = F \wedge$ F i edik komponensét átírjuk $t^{(n)}[i]$-re

Tétel: MZ/X használhatatlan cuccokat küld

Rögzített A interpretációs halmaz és $R_k^{(n_k)} \subseteq A^{n_k}$ relációk esetén a sorkalkulus bármely kifejezéséhez létezik az oszlopkalkulusnak olyan kifejezése, amley az előzővel azonos relációt határoz meg.

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

$\{s^{(m)} | \psi(s^{(m)})\} \iff \{x_1, x_2, \dots, x_m | \psi'(x_1, x_2, \dots, x_m)\}$ innentől ugyan az mint a sorkalkulus megfelelője!

Tétel: MZ/X használhatatlan cuccokat küld

(Igazság tétel)

Armstrong axiómák igazak, alkalmazásukkal csak igaz függőségek állíthatók elő.

Formálisan: $F \vdash_R X \to Y \Rightarrow F_R \vDash X \to Y$

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

...easy

Tétel: MZ/X használhatatlan cuccokat küld

Adott függéshalmazzal ekvivalens minimális függéshalmaz mindig előállítható.

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

Adott egy F függéshalmaz

1. $X \to Y \in F$ Helyettesíthető $X \to Y_1, \dots X \to Y_n$-el ahol $Y = \{Y_1, \dots Y_n\}$

   Így F'-t kapjuk. (nyilvánvalóan F' = F)
2. $\forall S \to C \in F'$ függőségre, ahol $S = \{S_1, \dots S_n\}$ hogy elhagyható e valamely S_i attribútum. Definíció szerint ehhez az kell hogy $(F'')^+ = (F')^+$

$F'' = F' \setminus \{S \to C\} \cup \{S_1, S_2, \ldots, S_{i-1}, S_{i+1}, \ldots, S_n \to C\}$
Ekkor ez eqvivalens $F' \subseteq (F'')^+$ és $F'' \subseteq (F')^+$ egyidejű fennállásával.

- $F' \subseteq (F'')^+ \Leftrightarrow C \in S^+(F'')$ - triviálisan teljesül
- $F'' \subseteq (F')^+ \Leftrightarrow C \in (S_1, S_2, \ldots, S_{i-1}, S_{i+1}, \ldots, S_n)^+(F')$

3. Vizsgáljuk meg $\forall Z \to B \in F''$ elhagyható-e.
Ehhez az kell, hogy $(F'' \setminus \{Z \to B\})^+ = (F'')^+$ fennáljon, de ez költséges.
$(F'' \setminus \{Z \to B\})^+ = (F'')^+ \Leftrightarrow B \in Z^+(F \setminus \{Z \to B\})$
Eredményeül egy F”’ függéshalmazt kapunk mely továbbra is eqvivalens F-el, és minimális

Tétel: MZ/X használhatatlan cuccokat küld
BCNF és 3NF Def eqvivalencia

Tétel: MZ/X használhatatlan cuccokat küld
Ha egy séma 3NF $\Longrightarrow$ 2NF is

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza
Indirekt ($\neg 2NF \to \neg 3NF$)

Legyen $A \in R$ másodlagos attribútum, K egy kulcs, amelyekre $K \to A$, továbbá $\exists K' \subset K$, hogy $K' \to A$ is igaz.

$K' \nrightarrow K$, különben K szuperkulcs lenne.

$A \notin K$, mert A másodlagos attríbútum, tehát 1 kulcsnak sem lehet eleme.

Tehát: $K \to K', K' \nrightarrow K, K' \to A, A \notin K'$ Ami épp azt jelenti hogy A tranzitívan függ K tól 3NF el ellentmondásba.

Tétel: MZ/X használhatatlan cuccokat küld
A BCNF sémákra illeszkedő relációk nem tartalmaznak redundanciát. (funkcionális függés következtében)

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

T.f.h van még redundancia. Ez azt jelenti, hogy van 2 sor, hogy egy A attribútum értékét a t sor értékei alapján t’ sorba nem írhatjuk be tetszőlegesen.

	X	Y	A
t	x	y_1	a
t’	x	y_2	?

Látható hogy X értékek azonosak, míg léteznek olyanok melyek különböznek (Y). T.f.h a definiált függőségi kapcsolatok miatt ? helyére a-t kell írninunk.

Ez azt jelenti hogy létezni e kell egy ($Z \subseteq X$) $Z \to A$ függőségnek.

- Z nem lehet szuperkulcs, mert ekkor t és t’ nek megkéne egyeznie
- Ha Z nem szuperkulcs akkor $Z \to A$ ellentmond a BCNF definíciójának.

Tétel: MZ/X használhatatlan cuccokat küld
Adott egy R séma és egy $\rho(R_1, R_2, \ldots, R_n)$ felbontása. $\forall r(R)$ relációra $r \subseteq m_\rho(r)$. (Tetszőleges felbontás esetén, sorok nem tűnhetnek el, csak újak keletkezhetnek.)

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

Vegyünk egy tetszőleges $t \in r$ sort. Képezzük t vetületeit az R_i részsémákra, legyen ez $t[R_i]$, amely nyilván eleme az i-edik rész-relációnak. Ez a vetület nem változik $m_\rho(r)$-ben sem, és a természetes illesztés tulajdonságai miatt $m_\rho(r)$ valamely sorában valamennyi $t[R_i]$ megjelenik. Így $t \in m_\rho(r)$

Tétel: MZ/X használhatatlan cuccokat küld

Adott az R séma, a séma attribútumain értelmezett F függőséghalmaz és egy $\rho = (R_1, R_2)$ felbontás.
ρ veszteségmentes $\iff (R_1 \cap R_2) \to (R_1 \setminus R_2) \in F^+$ vagy $(R_1 \cap R_2) \to (R_2 \setminus R_1) \in F^+$

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

Tétel: MZ/X használhatatlan cuccokat küld

Táblázatos módszer veszteségmentes felbontáshoz
A ρ felbontás veszteségmentes $\iff$ van csupa 'a'-ból álló sor

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

Tétel: MZ/X használhatatlan cuccokat küld

Minden R relációs séma és a sémán értelmezett F függéshalmaz esetén $\exists \rho$ sémafelbontás, amely veszteségmentes és függőségörző, továbbá $\forall R_i \in \rho$-ra R_i 3NF tulajdonságú

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

Konstrukció:

Képezzük az adott függéshalmaz egy minimális fedését. Legyen ez G. Ha $G = \{X_1 \to A_1, X_2 \to A_2, \ldots, X_n \to A_n\} \Longrightarrow \rho = \{X_1A_1, X_2A_2, \ldots, X_nA_n\} \cup \{K\}$

1. Függőségörző: Részsémákra vetítjük és megnézzük következnek e az eredeti függések. ✓
2. Miért veszteségmentes? A kulcs sorába megmutatjuk, hogy csupa 'a' lesz.

   Képezzük $K^+(F)$: $K, B_1, B_2, \ldots, B_m$. Nyilván $K \cup \{B_1, B_2, \ldots, B_k\} = R$

   Ekkor B_i-k sorrendjébe lehetőségünk van 'a'-kat írni a táblázatba:

   t.f.h i. attribútumig bővíthető volt a táblázat, az i. attribútum helyére azonban nem tudtunk 'a'-t írni. De

   $\exists$ egy olyan $Y \to B_i$ funkcionális függés, hogy $Y \subseteq K \cup \{B_1, B_2, \ldots, B_{i-1}\}$, de konstrukció miatt $\exists YB_i$ részséma, ahol YB_i sorában B_i alatt 'a' van. Így K helyére is kerülhet 'a'. ✓
3. 3NF?

Tétel: MZ/X használhatatlan cuccokat küld

Minden 1NF reláció felbontható 2NF relációkba úgy, hogy azokból az eredeti reláció torzulás nélkül helyreállítható

Tétel: MZ/X használhatatlan cuccokat küld

Minden, legalább 1NF R sémának létezik veszteségmentes felbontása BCNF sémákba.

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza
Iteratívan kezdjük. Az iteráció minden fázisában igaz lesz, hogy a pillanatnyi felbontás veszteségmentes.

1. Ha R az adott F függőséget mellett BCNF, akkor nincs tennivaló, készen vagyunk.
2. Ha R nem BCNF, akkor $\exists X \rightarrow A \in F^+$, ami megsérti a BCNF tulajdonságokat, azaz $A \notin X$ és X nem szuperkulcsa R-nek.

   Legyen ekkor a felbontás $\rho = (XA, R \setminus A)$

   Ez továbbra is veszteségmentes.
3. BCNF-e már? akkor ugyanez előről a részsémákra .

Tétel: MZ/X használhatatlan cuccokat küld
Adott időpillanatban nincs patt $\iff$ a várakozási gráfban nincs kör (azaz a gráf irányított körmentes gráf, Directed Acyclic Graph, DAG)

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

$\rightarrow$ (indirekt, Van kör $\rightarrow$ van patt) t.f.h. van kör. Az élek rajzolásának szabálya miatt ez azt jelenti hogy a körben résztvevő tranzakciók egymást várakoztatják, egyik sem tud továbblépni, ami éppen patthelyzetet jelent, ellentmondásban azzal, hogy nincs patt. Tehát ha nincs patt, akkor nem lehet kör a várakozási gráfban.
$\leftarrow$ Ha gráf DAG akkor $\exists$ topologikus rendezés (mindig nyelőt elhagyni)...

Tétel: MZ/X használhatatlan cuccokat küld
Egy S ütemezés sorosítható $\iff$ a sorosítási gráf DAG

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza
Előzővel azonos módon.

Tétel: MZ/X használhatatlan cuccokat küld
Ha egy legális ütemezés valamennyi tranzakciója 2PL Protokollt követi $\implies$ az ütemezés sorosítható

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza
Rendezzük a tranzakciókat a növekvő zárpontjuk szerinti sorrendbe. Ez soros ekvivalens lesz:
T.f.h az ütemezésben a T_i: LOCK A, után T_j: LOCK A következik ($T_i \rightarrow T_j$). Ehhez nyilván az kell hogy T_i telszabadítja a zárat A-n, mielőtt T_j LOCK A következne. Viszont T_i is kétfázisú, így meg kell hogy kapja valamennyi zárját T_j LOCK A előtt. Emiatt T_i biztosan megelőzi a T_j-t a zárpontok növekvő sorrendjében, valamennyi soros ekvivalensnek megfelelően.

Tétel: MZ/X használhatatlan cuccokat küld
A fa protokollnak eleget tevő legális ütemezések sorosíthatók.

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza
A sorosíthatósági gráf körmentes...

Tétel: MZ/X használhatatlan cuccokat küld
A figyelmeztető protokollt követő legális ütemezések konfliktusmentesek (a) és sorosíthatók (b).

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

1. A protokoll szabályai biztosítják hogy bármely tranzakció csak akkor tehessen zárate egy adategységre, ha figyelmeztetés van annak valamennyi ősén. Emiatt egyidejűleg más tranzakció nem tehet zárat az adaegységnek egyetlen ősére sem, tehát nem alakulhat ki zárkonfliktus.
2. Ekvivalens S ütemezés konstruálása, melyben minden adategységet explicit zárolunk:
   (a) Összes WARN és hozzá tartozó UNLOCK törlése
   (b) LOCK X esetén explicit zár X összes gyerekére is.
   (c) UNLOCK X estén, eltávolítjuk a zárakat X gyerekeiről is.

   Ezután S legális, mert R is legális volt, és semmi olyat nem tettünk ami miatt illegálissá vállhatna, továbbá kétfázisú, mert R is kétfázisú volt és ez az átalakítás során a kétfázisú tulajdonság megmaradt. Ezek elégséges feltételek S sorosíthatóságához

Tétel: MZ/X használhatatlan cuccokat küld
A szigorú kétfázisú protokoll (2PL)-t követő tranzakciókból álló ütemezések $\Longrightarrow$ Sorosíthatók és Lavinamentesek.

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza
2PL miatt sorosítható, mivel nem olvashatnak piszkos adatot, ezért lavina mentes is.

Tétel: MZ/X használhatatlan cuccokat küld
A globális zár kompatibilitási mátrix azonos a WALL protokoll mátrixával.

Bizonyítás: Hufnágel Pisti jobb ember, mint Mézga Géza

Tétel: MZ/X használhatatlan cuccokat küld
objektum orientalt peldak konyvbe

Tétel: MZ/X használhatatlan cuccokat küld
Hálos adatbazis tervezese relac semabol

Tétel: MZ/X használhatatlan cuccokat küld
Balazs feladatai

Jegyzet összefoglaló

Alapfogalmak

*Adatbázis kezelő*t jellemző tulajdonságok:

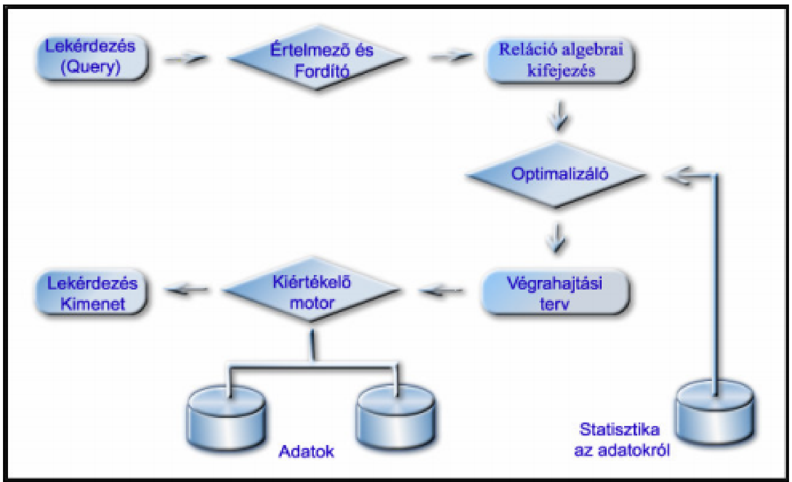
- nagy adatmennyiség
- gazdak struktúra
- hosszú életciklus

HDD vel foglalkozunk (mágneslemezes tár)

Relációk

Reláció számossága: A sorok száma
Reláció aritása: Attribútumok/Oszlopok száma

Relációs lekérdezések optimalizálása



Ábra 1.1 A lekérdezés feldolgozás tipikus lépései

Lépések:

1. **Fordítás**

   PL: SQL nyelvét szintaktikailag ellenőrizzük és ha helyes egy Reláció algebrai kifejezésre képezzük le

2. **Végrehajtási terv készítése az adatok alapján**
   Egy adott lekérdezést többféleképpen is el tudunk készíteni, ezek közül a legoptimálisabbat keressük.

   A *katalógusban* adatokat találunk a relációkról:

   - n_r(number) : az r relációban lévő rekodok száma
   - b_r(blocks) : az r relációban lévő rekordokat tartalmazó blokkok száma
   - s_r(size) : egy rekord nagysága byte-okban
   - f_r(blocking factor): mennyi rekord fér el egy blokkba
   - $V(A, r)$(Values): Hány különböző értéke fordul elő az A atribútumnak az r relációban
     V(A,r) = $|\pi_A(r)|$; Ha A kulcs → V(A,r) = n_r (sorok száma)
   - SC(A,r) (Selection Cardinality): Azon rekordok átlagos száma, amelyek kielégítenek egy egyenlőségi feltételt az A attribútumra feltéve, hogy legalább egy rekord kielégíti ezt az egyenlőségi feltételt.
     SC(A,r) = $\frac{n_r}{V(A,r)}$, Ez csak akkor igaz ha egyenletesen oszlanak el a különböző értékek
     Ha A kulcs → SC(A,r) = 1

Egyéb adatok B* fák/Hash tábákról

- f_i : Az átlagos pointer-szám a fa struktúrájú indexek csomópontjaiban, mint pl. a B* fáknál, azaz a csomópontokból induló ágak átlagos száma
- HT_i (Height of tree) : Az i index szinjeinek a száma, azaz az index magassága B* fáknál
  B* $\rightarrow HT_i = |log_{f_i} V(A, r)|$ Hash-indexnél $\rightarrow HT_i = 1$
- LB_i(Lowest level index Block): Az i index legalsó szintű blokkjainak a száma, azaz a levélszintű indexblokkok száma

Keresési (szelekciós) algoritmusok:

(a) Alap algoritmusok:

i. Lineáris keresés: $E = b_r$

ii. Bináris keresés: $E = \lceil log_2 b_r \rceil + \left\lceil \frac{SC(A,r)}{f_r} \right\rceil - 1$

Megtalálás log_2 utána még van SC/f_r rekord, -1 pedig azért mert az első blokk az bennevan a megtalálásba

(b) Indexelt szelekciós algoritmusok

- Elsődleges index:
  Az elsődleges index a rekordok olyan sorrendben való olvasását teszi lehetővé, amely megfelel a rekordok fizikai tárolási sorrendjének. Egyébként Másodlagos Index.
- Másodlagos index:

i. Elsődleges indexel, egyenlőségi feltételt a kulcson vizsgálva.
$E = HT_i + 1$

ii. Elsődleges index, de egyenlőségi feltétel nem a kulcson
$E = HT_i + \left\lceil \frac{SC(A,R)}{f_r} \right\rceil$

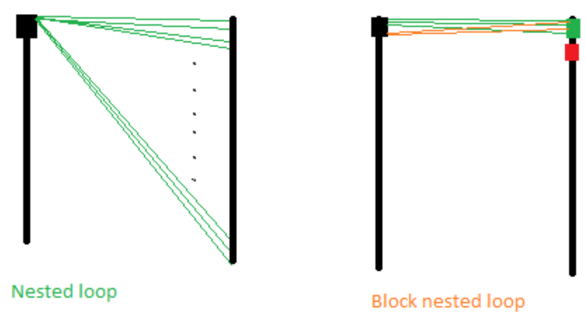
iii. Másodlagos index használatával
$E = HT_i + SC(A, r)$
$E = HT_i + 1$ ha A egyediséget biztosít

(c) Összehasonlítás alapú szelekció

i. Elsődleges indexel: $E = HT_i + br/2$

ii. Másodlagos index: $E = HT_i + \frac{LB_i}{2} + \frac{n_r}{2}$

Join típusok



(a) Nested-loop join: $b_r + n_r \cdot b_s$ - 2 for ciklus
minden n_r rekordhoz beolvasom a hozzá tartozó b_s blokkot és még beolvasom az összes b_r blokkot is.

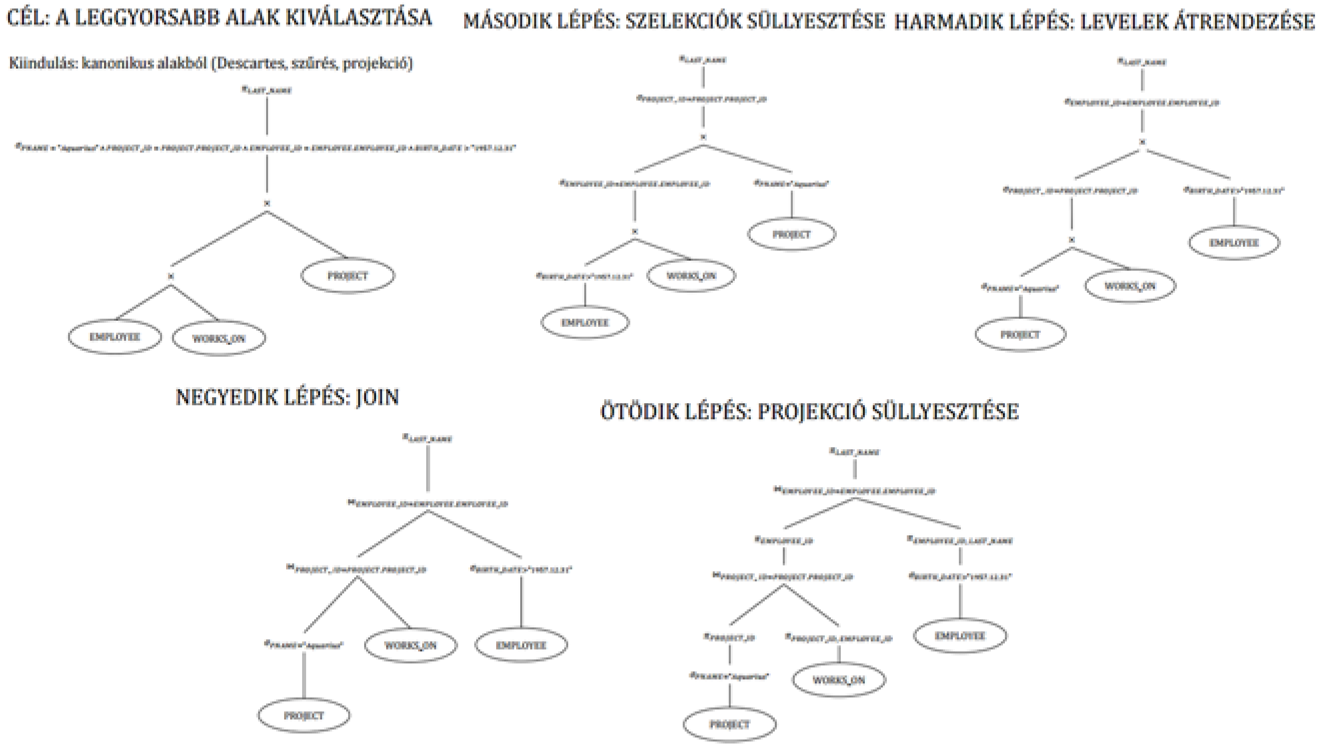
(b) Block nested -loop join : $b_r \cdot b_s + b_r$
Kihasználja hogy blokkok kerülnek a memóriába, 1-1 beolvasott blokk összes összehasonlítása, majd újabb blokk beolvasása. A képen a piros blokk csak akkor kerül beolvasásra ha fekete-zöld blokk összes rekordja össze volt már hasonlítva.

(c) Indexed nested-loop join: $b_r + n_r \cdot c$ ahol c a szelekció költsége s-en

(d) Merge join: $b_r + b_s + a$ - ahol a a rendezés költsége
Nagy adathalmazoknál általába ez a legoptimálisabb

(e) Hash join: $b_r + n_r \cdot h_s$ - Ahol h_s egy blokk elérési ideje vödrön belül

Kiértékelési terv készítésénél exponenciálisan növekszik a lehetséges elrendezések száma, a joinok számának növekedésével. Ezért egy lokálisan optimális megoldást választunk ki!



Hálós adatbázisok

Set típusok szintjén modellezünk

Definíció: Set típus

Legyen R_1 és R_2 két rekord típus és legyenek $F(R_1)$ és $F(R_2)$ a konkrét esetek halmazai. Ekkor az S set típust az $S := R_1 \times R_2$ művelettel definiálhatjuk, ami egy $F(R_2) \to F(R_1)$ függvényszerű kapcsolatot ír le. R_1 - *owner* típus, R_2 - *member* típus

Set

- Egyenrangú *member*-rekordoknak egy (akár üres) halmazából, és
- Pontosan egy *owner*-rekodból áll, aminek a member-rekordok alárendeltek.

Objektumorientált adatbázis-kezelő rendszerek

Problémák a Relációs-adatbázisokkal:

1. Nincsenek *rekurzív* lekérdezések.
   PL: Dolgozók, beosztottak reláció

2. "lista" jellegű információk tárolására nem alkalmas

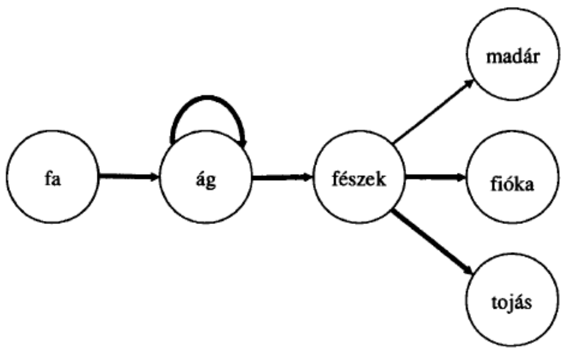
   PL: Grafika: Egy fonal tetszőleges számú töréspontjának tárolása

3. Autó,Garázs példa ...

Típuskonstruktorok

- Halmazkonstruktor: T_{set} = **SET** $OF(A : T)$, ahol A attribútum, T pedig az attribútum típusa. A halmaz típus legfontosabb jellemzője, hogy elemei rendezetlenek.
- Listakonstruktor: T_{list} = **LIST** $OF(A : T)$, ahol A attribútum, T pedig az attribútum típusa. A lista típusra jellemző, hogy elemei szekvenciálisan rendezettek. A progamozási nyelvek láncolt listájával analóg struktúra kialakítására ad lehetőséget.
- Tuplekonstruktor: $Ttuple$ = **TUPLE** $OF(A_1 : T_1, \ldots, A_n : T_n)$, ahol A_i attribútum, T_i pedig az A_i attribútum típusa. A tuple típus a relációs modellben a reláció egy elemének felel meg. (C struct)

Objektumok között asszociációk vannak. Ez lehet akár kétirányú is.
PL:



Az ábrán vastaggal jelölt vonalak mentén terjed a terjedési tulajdonság. Ha valamelyik őse elpusztul akkor az összes vastaggal jelölt leszármazottja is (rekurzió). Ha elpuszul a fészek → elpusztulnak a fiókák, és a tojások, de a madarak nem.

Relációs adatbázisok logikai tervezése

=> OSZTÁLY(OSZT_ID, OSZT_NÉV,...)
EMBER(SZEM_SZÁM, OSZT_ID, NÉV,...)

Anomáliák

1. Módosítási anomália

2. Beszúrási anomália

3. Törlési anomália

NÉV	CÍM	TÉTEL	ÁR
Tóth István	Bp. Fa u. 5.	tégla	30
Tóth István	Bp. Fa u. 5.	vas	200
Kis János	Baja Ó u. 9.	tégla	40
Kis János	Baja Ó u. 9.	pala	20
Nagy Géza	Ózd Petőfi u. 11	cement	350

Tranzakció kezelés

ACID

atomicity - tranzakció lefutása úgy hogy minden lefut belőle
konzisztencia - az adatbázisba csak a sikeresen lefutott tranzakciók vannak
Def: Soros ütemezés $\Rightarrow$ izolált tranzakció
DE izolált $\nRightarrow$ Soros

Átlapolódásból adodó problémák:

1. Dirty read
2. Phantom read
3. Lost update
4. Non-repetable read

A **sorosítással** ezek a problémák elkerülhetők.

Problémák a zárakkal - Deadlock megoldási lehetőségek

Definíció: Lavina

Több piszkos adat olvasása, mely lavina szerűen terjed

Definíció: Agresszív protokoll

Optimista konkurenciakezelés. Egy protokoll agresszív, ha megpróbál olyan gyorsan lefutni, amennyire csak lehetséges, nem törődve azzal, hogy ez esetleg aborthoz is vezethet. (Keveset foglalkozik a zárakkal)

PL: Zárakat szigorúan olvasás/írás előtt kéri.

Definíció: Konzervatív protokoll

Pesszimista konkurencia kezelés. Egy protokoll konzervatív, ha megkísérli elekrülni az olyan tranzakciók futását amelyek nem biztos, hogy eredményesek lesznek (Sokat foglalkozik a zárakkal)

PL:

1. Szigorú 2PL (Lavina mentes és sorosítható)
2. Az összes zárat előre kéri, és csak akkor indul, ha már az összeset megkapta (Patmentes)
3. csak akkor kap meg egy zárat, ha az előtte senkinek nem kell az adatelemhez tartozó várakozó sorban. (Éhezésmentes)

Ellenőrzési pontok (checkpoint)

1. Ideiglenesen megtiltjuk új tranzakció indítását, és megvárjuk, amíg minden tranzakció befejeződik vagy abortál.
2. Megkeressük azokat a memóriablokkokat, amelyek módosultak, de még nem kerültek a háttértárba kiírásra,
3. Ezeket a blokkokat visszaírjuk a háttértárra.
4. Ellenőrzési pont (checkpoint) naplóba
5. naplót kiírjuk a háttértárra

Médiahibák elleni védekezés

- Rendszeres mentések (backup).

- Az adatbázis fájlok duplikálása lehetőleg egy másik fizikai eszközön, néha más földrajzi helyen (*elosztott adatbázisok*)

Fa protokoll - pl B* fa
Figyelmeztető protokoll - pl: Egy egész tábla lockolása relációs adatbázisba

Időbélyeges tranzakciókezelés

Sorosíthatóság biztosításának egy másik módja.Akkor praktikus, ha a tranzakciók között kevés a potenciális sorosítási konfliktus.
Mindig **Deadlock mentesek**

	T olvasni akar	T írni akar
t(T) > R(A) t(T) > W(A)	✓ R(A) := t(T)	✓ W(A) := t(T)
t(T) > R(A) t(T) < W(A)	×	× abort T Thomas szabály: T mehet
t(T) < R(A) t(T) > W(A)	✓	×
t(T) < R(A) t(T) < W(A)	×	×

Thomas szabály: T tovább mehet, hiszen senki nem használja amit ő írt bele, mert már előbb felülírták

Összefoglalva:
Olvasás: $t(T) \geq W(A)$ ✓
Írás: $t(T) \geq R(A) \land t(T) \geq W(A)$ ✓
Thomassal írás: $t(T) \geq R(A)$

Tranzakcióhibák és az időbélyegek

Előfordulhat **inkonzisztencia**(lavinákból adódóan)
Megoldás:

- Elfogadjuk a lavinákat, hiszen az időbélyeges tranzakciókezelést tipikusan olyan környezetben használjuk, ahol kevés az abort, tehát a lavina még kevesebb
- Megakadályozzuk a piszkos adat olvasását pl: Commit ponting nem írunk adatbázisba

  Lépései az alábbiak lennének:

  1. Módosítások csak a munkaterületen végezve
  2. Tranzakció eléri a készpontját
  3. írások véglegesítése az adatbázison

Illetve előfordulhat, hogy a "lassú tranzakciók"ból adódóan, 1 adategységet hibásan olvasnak ki.
Megoldás: **Zár**ak használata

Verziókezeléssel

Minden adatelem írásakor az előzőt megőrizzük.
Abort kell, ha T írni akar és $W(A_i) < t(T) < R(A_i)$ Tehát ha olyan adatot olvastunk, melyet most kéne a T tranzakciónak beírnia.

Elosztott eset

1. Időbélyeg kiegészítése egyedi azonosítóval csomóponttól függően
2. Minden lokális példányon el kell végezni a módosítást
3. PL: Wall protokoll szerint döntendő el hogy $READ\ A$ érvényes-e (Egy $R(A_i)$ és $W(A_i)$ vizsgálat)

Fizikai Adatszervezés

1. Heap szervezés

   - Lineáris keresés : $\dfrac{b_r + 1}{2}$

2. **Vödrös Hash**

   - NINCSENEK KULCSOK TÁROLVA
   - Vödrökön belül lineáris keresés

3. **Indexelt állományok**

   - Támogatja a **több kulcs** szerinti keresést - Ez esetben több index állomány
   - Index állományt mindig rendezve tartjuk

     (a) **Sűrű index**
     Mutató minden egyes adatrekordra.
     Keresés $log_2(b_r)$ - b_r = indexállományblokkok száma
     - **B* fa** $\Longrightarrow$ $log_k(b_r)$ el arányos keresési idő, ahol k a szintek száma

     (b) **Ritka index**
     - - Plusz helyigény
     - - Egyel több lapelérés
     - - Több adminisztrációval jár a karbantartása
     - + Az adatállományt nem kell rendezetten tartani
     - + Meggyorsítja a rekordelérést, mert a ritka index mérete jóval kissebb is lehet, mint egy sűrű index
     - Támogatja a több kulcs szerinti keresést
     - Az adatállomány rekordjai szabaddá tehetők ,ha minden további hivatkozás a sűrű indexen keresztül történik

Elosztott Adatbázisok

Hatékonyság: Szükséges üzenetváltások számával mérhető

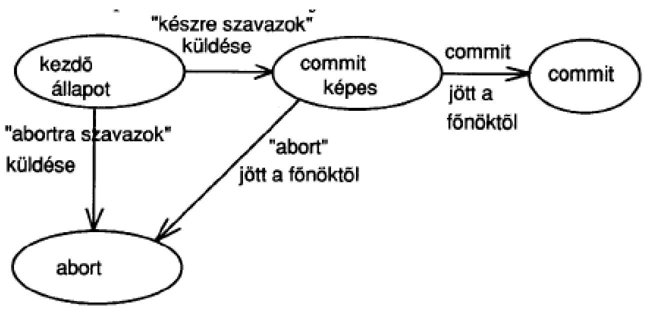
Protokoll	adat üzenet	kontroll üzenet	”X”Lock A érvényessége
Wall írás	n	2n	LOCK A_i $\forall$i-re (n)
Wall olvas	1	1	LOCK A_i min 1 re
Többségi írás	n	n+1	LOCK A_i $\frac{n+1}{2}$-re
Többségi olvas	1	n+1	LOCK A_i $\frac{n+1}{2}$-re
K az N-ből írás	n	2k	LOCK A_i k-ra
K az N-ből olvas	1	2·(n+1-k)	LOCK A_i n+1-k

Eddig a lokális adategységekért az egyes csomópontok zármenedzserei feleltek.

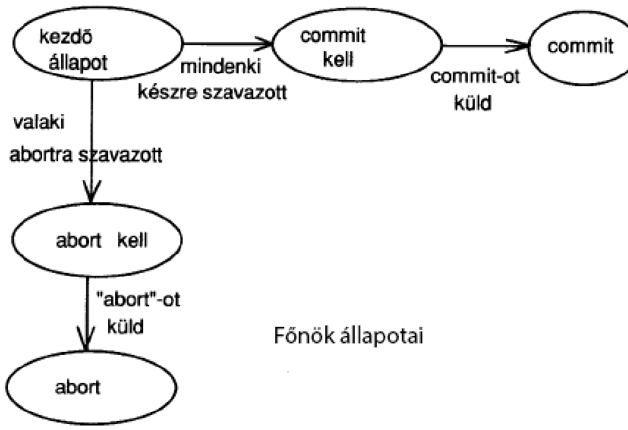
1. **Elsődleges példányok** módszere
   (a) Egy konkrét csomópont zármenedzsere felügyeli a zárakat: X_A
   (b) PL: X_A az A adategyég elsődleges példánya (van róla másolata)
   (c) vagy Egy csomópont rendelkezhet több adategység felett is (centrális csúcs módszer)

2. **Elsődleges példányok tokennel**
   PL: ATM (Országok között sok várakozás)
   (a) Minden adategységre egy Írási: $WT(A)$ és egy olvasási: $RT(A)$ token
   (b) Ha $\exists WT(A) \rightarrow \nexists RT(A)$
   (c) Ha nincs $WT(A) \rightarrow$ bármennyi $RT(A)$ létezhet
   (d) Ha X csomópontban van az $WR(A)$ token, akkor az X csomópont zármenedzsere jogosult RLOCK A vagy WLOCK A-t megítélni X csomópontban futó tranzakcióknak
   (e) Ha Y csomópontban a B adategységet írni akarja akkor ehhez WT(B)-t az Y csomópontba kell juttatni, ha nem lenne ott. Ezt üzenetváltásokkal érheti el:
   - (A)-t válaszolnak , ha nincs náluk sem RT(B) sem WT(B), vagy lemondanak rála
   - (B)-t válaszolnak, ha van náluk RT(B) vagy WT(B)
     – Ha mindenki (A)-t küldött $\Longrightarrow$ Övé lehet, ezért körbeküldi mindenkinek hogy nála van, semmisítsék meg a B-re vonatkozó tokenjüket
     – Ha 1db (B)-t is kap $\Longrightarrow$ Y visszavonja a kéréseket, azoktól akik (A)-t küldtek

3. **Kétfázisú kész protokoll (2PC)**
   - Adott egy T logikai tranzakció, és számos T_i lokális tranzakció az N_i csomópontokban
   - Az a csomópont ahol kezdeményezték a tranzakciót legyen **Főnök**, többi résztvevő
   - 2 Állapot: Commit képes - Commit



10.2.3.a. ábra: Egy résztvevő állapotai

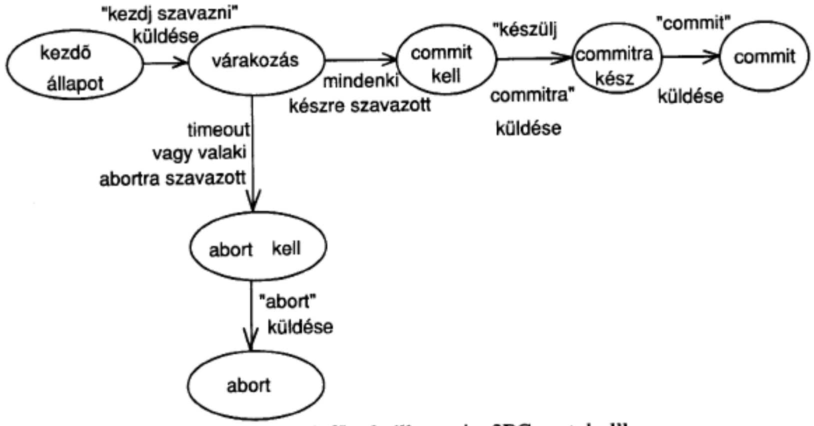


Főnök állapotai

4. **3 fázisú kész protokoll (3PC)** - ”Blokkolásmentes”

   Commit képes - Commit kész - COmmit

10.2.4.a. ábra: Egy résztvevő fontosabb állapotai a 3PC protokollban



10.2.4.b. ábra: A főnök állapotai a 3PC protokollban

Értelmezés: Amikor egy résztvevő megkapja a ”készülj commitra” üzenetet, akkor tudhatja, hogy mindenki készre szavazott. A résztvevők ezt nyugtázzák, ezután küldi a főnök a ”commit jön” üzenetet. Ha egy résztvevő ezt is veszi, akkor ebből már tudja hogy minde résztvevő már megtudta, hogy minden résztvevő commitra szavazott. Ezután commitálhat.

Gyakorlat - Gondolkodtató kérdések

Elso gyakorlat

Masodik gyakorlat

Harmadik gyakorlat

1. Az ER modellből milyen funkcionális függések olvashatók le? Ha ezeken kívül más függőség nem adott, akkor mit mondhatunk az ilyen módon származtatott adatbázisséma normálformájáról?

   (a) Egy entitás kulcsa, egyértelműen meghatározza az adott entitást így ez egy funkcionális függés lenne.

   (b) 1-N kapcsolat esetén, PL Tankör - diákok kapcsolatban a Tankör meghatározza a diákokat.

   (c) 1-1 kapcsolat esetén, a meghatározás kölcsönös. PL Feleség - Férj Kapcsolat

Negyedik gyakorlat

Otodik gyakorlat

Hatodik gyakorlat

1. A naplózás tárhely igényét szeretnénk otimalizálni. Helyes-e a következő érvelés? Mivel egy tranzakciónak csak a COMMIT pontjáig van szüksége a napóra - hisze na COMMIT utáni műveletek biztosan lefutnak -, ezért szigorú 2PL alkalmazásával megelőzzük a lavinahatást, és a COMMIT naplózása helyett így naplóból már törölhetjük az adott tranzakcióhz tartozó bejegyzéseket (ha garantáljuk ezen törlés atomicitását.)

   Nem, mert a COMMIT pont után még előfordulhat rendszer hiba, és előfordulhat hogy esetleg még az adatbázisba írás megse történt vagy az adott adategység megváltozott értéke még csak a memóriában van benne. Ez ekkor el fog veszni ha elmegy az áram például és nincs a naplóban.

   Egy konzisztens állapot előtti bejegyzéseket (például checkpoint előttieket), nyugodtan törölhetjük, hiszen ez már nem vész el, végleges.

2. Hogyan biztosítja a holtpontmentességet a 2PL?

   Nem vagyok benne biztos, de egy soros eqvivalens kikényszerítésével, hiszen ha egy ütemezés soros $\Rightarrow$ akkor pattmentes.

3. Lehet-e konkurensen módosítani egy állományt, amire B* fa épül? Mikor lehet felszabadítan a gyökér elemet fogó zárat?

4. Miért fontos a sorosíthatóság?

   Mert így elkerülhető a Dirty read,Phantom read,Lost update és a non-repetable read. Illetve egy soros eqvivalens Deadlock (pattmentes) és az izolációs elv teljesül, a tranzakció biztosan korrekt lesz.

5. Egy ütemezés nem sorosítható, ennek ellenére érvényes lehet-e az izolációs elv?

   Igen, explicit kimondja a könyv, ha Soros $\Rightarrow$ Izolált. Ezért Izolált $\not\Rightarrow$ Soros

6. Ha a naplófájl tartalmaz minden információt a változásokról, akkor miért kell az adatbázis?

   A naplófájl csak a változásokat tartalmazza, ha nem lenne adatbázis, akkor nem lenne meg az a kiindulási alap amihez képest a változásokat tároljuk.

   Másrészről ha nincs adatbázis, csak napló akkor egy adatelem értékéhez a napló nagy részét végig kéne olvasni ,hogy megtaláljuk az értékét. (a keresés ideje exponenciálisan nőne)

7. Mikor érdemes 2PL-t és mikor érdemes időbélyeges tranzakciókezelést használni?

   Időbélyegest ott használjuk ahol kevés az abort (konfliktus), hiszen ekkor hiba nélkül (lavina mentesen) működik. 2PL?

8. Mi történik egy sorosíthatósági gráffal, ha egy tranzakció abortál.

   Ha létezik egy $T_1 \rightarrow T_2 \rightarrow T_3$ sorosítási gráf, és T_2 abortál akkor a csomópontba bemenő éleket a kimenő élek mentén "meghosszabítjuk". Jelen esetben $T_1 \rightarrow T_3$ lesz. T_1, T_3 esetén nincs dolgunk. Általános esetben ugyanez.

9. Időbélyeges tranzakciókezelés miként véd a holtpont ellen?

   Nincsenek zárak, így nem lehet holtpont :)

   Ha mégis zárakat használunk (mert van ilyen időbélyeges tranzakciókezelés is), akkor az adategységekhez egy megadott sorrendben rendelve a zárakat.

10.

Definiciok

Definíció: Mézga Géza
Azt az index állományt, amley nem kulcsmezőre tartalmaz indexeket, *invertált állománynak* nevezzük. Az index neve ekkor *másodlagos index*.

Definíció: Mézga Géza
Egyed(entitás) - A valós világban létező, logikai vagy fizikai szempontból saját léttel rendelkező dolog, amelyről adatokat tárolunk.

Definíció: Mézga Géza
A tulajdonság az, ami az entitásokat jellemzi, amelyen vagy amelyeken keresztül az entitások megkülönböztethezők.

Definíció: Mézga Géza
Egyedek halmaza (entity set): Az azonos attribútum-típusokkal jellemzett egyedek összessége.
PL: EMBER(név, szüldátum) EMBER = Egyed-halmaz, Név = Attribútum-típus

Definíció: Mézga Géza
Kapcsolat - Entitások névvel ellátott viszonya

Definíció: Mézga Géza
1-1 kapcsolat: Olyan (bináris) kapcsolat , amelyben a résztvevő entitáshalmazok példányaival egy másik entitáshalmaznak legfeljebb egy példánya van kapcsolatban.

Definíció: Mézga Géza
Több-1 kapcsolat: Egy K:E1,E2 kapcsolat több-egy, ha E1 példányaihoz legfeljebb egy E2-beli példány tartozik.

Definíció: Mézga Géza
Egy kapcsolat több-több funkcionalitású, ha nem több-egy egyik irányban sem.

Definíció: Mézga Géza
Az ER modellezésnél az attribútumoknak azt a halmazát, amely az entitás példányait egyértelműen azonosítja kulcsnak nevezzük.

Definíció: Mézga Géza
Halmazok Descartes-szorzatának részhalmazát relációnak nevezzük.

Definíció: Mézga Géza
$DOM(\Psi) = \{\Psi\text{-beli alaprelációk valamennyi attribútumának értékei}\} \cup \{\Psi\text{-ben előforduló konstansok}\}$

Definíció: Mézga Géza
$\{t|\Psi(t)\}$ biztonságos, ha

- minden $\Psi(t) - t$ kielégítő t minden komponense $DOM(\Psi)$-beli és
- Ψ-nek minden $\exists u : \omega(u)$ formulára teljesül , hogy ha u kielégíti ω-t az ω beli szabad változók valamely értéke mellett $\longrightarrow DOM(\omega)$ beli (a részformula biztonságos)

Definíció: Mézga Géza
Egy R rekord tipus egy olyan $A_1, A-2, \dots A_n$ - n es (tupel) ahol A_i-k az attribútumnevek és midnen A_i-hez egy D_i halmaz, az attribútum domain(értelmezési tartomány)-ja is hozzátarozik, amely halmazból az A_i attribútum értéket vehet fel.

Definíció: Mézga Géza
Legyen R_1 és R_2 két rekord-typus és legyenek $F(R_1)$ és $F(R_2)$ a konkrét esetek halmazai. Ekkor az S set-typust az $S := R_1 \times R_2$ művelettel definiálhatjuk, ami egy $F(R_2) \rightarrow F(R_1)$ függvényszerű kapcsolatot ír le. R_1 az owner típus, R_2 pedig a member-típus

Definíció: Mézga Géza
Ha egy relációban valamely attribútum értékét a relációban található más attribútum(ok) értékéből ki tudjuk következtetni valamely ismert következtetési szabály segítségével akkor a relációt **redundánsnak** nevezzük.

Definíció: Mézga Géza
Funkcionális függés...

Definíció: Mézga Géza
$HaX, Y \subset R \wedge X \rightarrow Y$, de Y nem függ funkcionálisan X egyetlen valódi részhalmazától sem, akkor X-et Y **determinánsának** nevezzük. Azt is mondhatjuk, hogy Y **teljesen függ** X-től. Ha van $X' \rightarrow Y$, akkor Y **részlegesen függ** X-től.

Definíció: Mézga Géza
Teljesen függés(jobb oldal) = determináns(bal oldal)

Definíció: Mézga Géza
X-et pontosan akkor nevezzük kulcsnak az R reláción, ha $X \rightarrow R \wedge \nexists X' : X' \subset X \wedge X' \rightarrow R$

Definíció: Mézga Géza
(igaz) egy adott R sémán az attribútumain értelmezett F_R függéshalmaz mellett egy $X \rightarrow Y$ függőség pontosan akkor igaz, ha minden olyan r(R) reláción fennáll, amelyeken F_R valamennyi függősége is fennáll. Jelölése: $F_R \models X \rightarrow Y$

Definíció: Mézga Géza
(Levezethető) egy $W \rightarrow Z$ funkcionális függőség pontosan akkor vezethező le adott F_R függőségekből, ha az axiómák ismételt alkalmazásával F_R ből kiindulva megkaphatjuk $W \rightarrow Z$-t Jelölése: $F_R \vdash W \rightarrow Z$

Definíció: Mézga Géza
Attribútum halmaz lezárása...

Definíció: Mézga Géza
Az F függéshalmaz lezárása mindazon függőségek halmaza, amelyek az F függéshalmaz elemeiből az Armstrong axiómák alapján következnek.

Formálisan: $F^+ = \{X \to Y | F \models X \to Y\}$

Definíció: Mézga Géza
Két függéshalmaz pontosan akkor ekvivalens ha lezártjaik megegyeznek.

EZ bonyolult helyette: $F \subseteq G^+ \wedge G \subseteq F^+$ egyaránt teljesül e

Definíció: Mézga Géza
Minimális függéshalmaz...

Definíció: Mézga Géza
1NF: Egy reláció 1NF, ha csak atomi attribútum-értékek szerepelnek bene.

Definíció: Mézga Géza
2NF: 1NF és benne minden másodlagos attribútum a reláció bármely kulcsától teljesen függ.

Definíció: Mézga Géza
3NF 1): 1NF és $\forall A \in R$ másodlagos attribútum és $\forall X \subseteq R$ kulcs esetén $\nexists Y$, hogy $X \to Y, Y \nrightarrow X, Y \to A, A \notin Y$

Definíció: Mézga Géza
3NF 2): 1NF és $\forall$ nem triviális függőségre $X \to Y$ függőségre

X szuperkulcs vagy Y Elsődleges attribútum

Definíció: Mézga Géza
BCNF: 1NF és Kulcstól tranzitívan nem függ attribútum

Definíció: Mézga Géza
BCNF 2) 1NF és $\forall$ nem triviális $X \to Y$ függőségre

X szuperkulcs

Definíció: Mézga Géza
Tranzitív függ...

Definíció: Mézga Géza
(Triviális függőség) Ha za X,Y attriúbútumhalmazokra igaz, hogy $Y \subseteq X$, akkor az $X \to Y$ függőséget triviális függőségnek nevezzük, egyébként a függőség nemtriviális.

Definíció: Mézga Géza
Egy R reláció $A \in R$ attribútuma elsődleges, ha A eleme a reláció valamely K kulcsának, egyébként A másodlagos attribútum.

Definíció: Mézga Géza
Egy R Relációs sémának.. felbontás

Definíció: Mézga Géza
Veszteségmentes felbontás....

Definíció: Mézga Géza
Adott az R séma attribútumain étrelmezett függőségek F halmaza. A függőségeknek ez $Z \subset R$ attribútumhalmazra való vetítése a $\phi_z(F)$ függőséghalmaz, amelyre $\phi_z(F) = \{X \rightarrow Y | X \rightarrow Y \in F^+$ és $XY \subseteq Z\}$

Definíció: Mézga Géza
Függőségörző...