

Számítógépes alkalmazások Adatbázis-kezelés

Soós Sándor, egyetemi adjunktus

Nyugat-magyarországi Egyetem
Simonyi Károly Műszaki, Faanyagtudományi és Művészeti Kar
Informatikai és Gazdasági Intézet

E-mail: soossandor@inf.nyme.hu

Sopron, 2015.



Tartalomjegyzék

- 1 Adatbázis-kezelés
 - Az adatbázis fogalma
 - Tranzakciókezelés

- 2 Adatbázis-kezelő rendszerek
 - Relációs adatmodell
 - Funkcionális függőség



Miről lesz szó a mai órán?

- adatok, adattáblák, adatbázisok
- tranzakciók, tranzakciókezelés
- zárolás, lock
- relációs adatmodell
- relációs algebra
- relációs adatbázisok
- funkcionális függőség
- redundancia
- anomáliák
- dekompozíció
- adatbázis-tervezés



Outline

- 1 Adatbázis-kezelés
 - Az adatbázis fogalma
 - Tranzakciókezelés
- 2 Adatbázis-kezelő rendszerek
 - Relációs adatmodell
 - Funkcionális függőség



Adat és adatbázis

- Az Excel-lel is kezeltünk adatokat
- Az adatbázis volt?
- Az Excel adatbázis-kezelő?
- Mikor válik az adat adatbázissá?



Az adatbázis

- Vizsgálandó szempontok:
 - Méret
 - elfér az operatív memóriában
 - nem fér el az operatív memóriában
 - Bonyolultság
 - egyszerű, egymástól független adatsorok (Excel)
 - kapcsolatok vannak az adatok között
 - Speciális adattípusok kezelése
 - kép
 - hang
 - videó



Az adatbázis

Mit nevezünk adatbázisnak?

- Az adatbázis azonos minőségű (jellemzőjű), többnyire strukturált adatok összessége, amelyet egy tárolására, lekérdezésére és szerkesztésére alkalmas szoftvereszköz kezel
- Az adatbázis nem azonos az adatbázis-kezelővel!



Adatbázis-kezelő

Mit várunk el egy adatbázis-kezelő rendszertől?

- 1 Tegye lehetővé új adatbázisok létrehozását, az adatbázis logikai struktúrájának (séma) leírását egy erre alkalmas nyelven (adatdefiníciós nyelv)
- 2 Tegye lehetővé az adatok biztonságos tárolását
- 3 Tegye lehetővé az adatok felvitelét, módosítását és törlését (adatmanipuláció nyelv)
- 4 Tegye lehetővé az adatok lekérdezését (lekérdező nyelv)
- 5 Legyen lehetőség a különböző műveletek elvégzésének lehetőségét jogosultsági szintekhez kötni
- 6 Mindez megbízhatóan és hatékonyan működjön akkor is, ha nagy mennyiségű (elvileg „korlátlan”) adatról van szó
- 7 Mindez megbízhatóan működjön akkor is, ha egyszerre több felhasználó szeretne műveleteket végezni



Egy tipikus példa: Bank

- Adatelemek:
 - ügyfelek: név, cím, jogosultságok
 - bankszámlák: folyószámlák, hitelszámlák, egyenlegek
 - az ügyfelek és a számlák között fennálló kapcsolatok
 - tranzakciók
- Műveletek:
 - új ügyfél létrehozása
 - új számla nyitása
 - számla megszüntetése
 - pénz befizetése
 - pénz kivétele
 - pénz utalása egyik számláról a másikra



Kritikus művelet: Pénz felvétele

- Van egy bankszámla 100 000 Ft-os egyenleggel
- A számlának két tulajdonosa van Zoli és Kati, mindkettőnek van joga pénzt befizetni és kivenni is
- Egy időben bemegy a két tulajdonos két különböző bankfiókba, mindkettő fel akar venni 80 000 Ft-ot
- Mi történik a rendszerben?
 - 1 Zoli belép a budapesti bankfiókba, kér 80 000 Ft-ot
 - 2 Az ügyintéző megnézi a számlát, úgy látja van rajta ekkora összeg, elkezd a kifizetést, megírja a pénztárbizonylatot
 - 3 Ekkor Kati belép a zalaegerszegi bankfiókba, kér 80 000 Ft-ot
 - 4 A zalaegerszegi ügyintéző ellenőrzi a számlát, az egyenleg 100 000 Ft, megírja a pénztárbizonylatot
 - 5 A budapesti ügyintéző kifizeti a pénzt, csökkenti a számla egyenlegét 20 000 Ft-ra
 - 6 A zalaegerszegi ügyintéző is fizet, csökkenti a számla egyenlegét –60 000 Ft-ra
 - 7 Mi történt? Miért?

Outline

- 1 Adatbázis-kezelés
 - Az adatbázis fogalma
 - Tranzakciókezelés
- 2 Adatbázis-kezelő rendszerek
 - Relációs adatmodell
 - Funkcionális függőség



Tranzakciókezelés

A probléma:

- A probléma oka, hogy megengedtük, hogy bizonyos kritikus műveletek végrehajtása közben mások is végrehajthódjanak, olyanok is, amelyek befolyásolhatják az első művelet eredményét
- Konkrétan: két művelet tudta módosítani a számla egyenlegét egyidőben

A megoldás:

- 1 Definiáljunk olyan elemi műveleteket, amelyek nem szakíthatók félbe, vagy a teljes folyamat lezajlik, vagy nem történik semmi (Tranzakció, Transaction)
- 2 Kritikus műveletek elvégzésének idejére zároljuk az adatbázis egyes részeit, hogy másik folyamat ne férhessen hozzá (Zárolás, Lock)



Az előző pénzfelvétel így néz ki biztonságosan:

- 1 Zoli belép a budapesti bankfiókba, kér 80 000 Ft-ot
- 2 Az ügyintéző megnézi a számlát, úgy látja van rajta ekkora összeg, ezért zárolja a számlát és elindít egy pénzfelvételi tranzakciót
- 3 Ekkor Kati belép a zalaegerszegi bankfiókba, kér 80 000 Ft-ot
- 4 A zalaegerszegi ügyintéző ellenőrzi a számlát, az egyenleg 100 000 Ft, de a számla zárolt, ezért nem indíthat el egy pénzfelvételi tranzakciót, várakozik
- 5 A budapesti ügyintéző kifizeti a pénzt, csökkenti a számla egyenlegét 20 000 Ft-ra, ezzel sikeresen befejeződik a tranzakció, feloldjuk a zárolást
- 6 A zalaegerszegi ügyintéző észleli, hogy újra szabad a számla, de az egyenleg már csak 20 000 Ft
- 7 Megmondja az ügyfélnek, hogy nincsen 80 000 Ft a számlán

Tranzakció

Tranzakció:

- Az idegen szavak szótárában:
 - bankügylet
- Az informatikában:
 - olyan több lépésből álló művelet sor, ami csak egységes egészsként hajtható végre
 - vagy lefut az egész tranzakció, vagy nem történik semmi
 - ha a tranzakció elkezdődik, de valamilyen okból nem tud szabályosan befejeződni, akkor a rendszer visszavonja az eddig végrehajtott lépéseket, visszaállítja a tranzakció elindulása előtti állapotot
 - nem csak banki rendszerekben és nem is csak adatbázis-kezelő rendszerekben használjuk



Tranzakciókezelés

```
TRANSACTION BEGIN
...
    a tranzakció lépései
...
if (nincs hiba) then
    COMMIT
else
    ROLLBACK
endif
```

- a programozónak ennyit kell tennie
- a többi elintézi a tranzakciókezelő rendszer



Zárolás – Lock

- Adott egy megosztott erőforrás, amit több felhasználó szeretne használni, de egy időben csak az egyik férhet hozzá
- Pl. közös nyomtató, egy fájl a fájlszerveren, egy rekord az adatbázisban, ...
- Minden felhasználó a következőképpen használja az erőforrást:

...

ERŐFORRÁS LOCK

...

az erőforrás használata

...

ERŐFORRÁS UNLOCK

...



Zárolás – Lock

- nagyon fontos, hogy ha egyszer zároltunk egy erőforrást, akkor azt fel is szabadítsuk
- ha elmarad, akkor senki nem fog tudni hozzáférni az erőforráshoz, rendszergazdai beavatkozásra van szükség
- kombináljuk a két megoldást, egy tranzakcióban használjuk az erőforrásokat

```
TRANSACTION BEGIN
    ERŐFORRÁS LOCK
        az erőforrás használata
    ERŐFORRÁS UNLOCK
if (nincs hiba) then
    COMMIT
else
    ROLLBACK
endif
```



Adatbázis-kezelő rendszerek típusai

- hierarchikus
- hálós
- relációs
- objektumorientált
- deduktív

Napjainkban a relációs adatmodell és az erre épülő adatbázis-kezelő rendszerek a legelterjedtebbek

Ezzel fogunk megismerkedni



Outline

- 1 Adatbázis-kezelés
 - Az adatbázis fogalma
 - Tranzakciókezelés
- 2 Adatbázis-kezelő rendszerek
 - Relációs adatmodell
 - Funkcionális függőség



Relációs adatmodell

- E. F. Codd:
A relational model of data for large shared data banks.
Communications of the ACM, 13 (1970), 377-387.
- A felhasználó számára táblázatok (relációk) formájában jeleníti meg az adatokat
- A relációkkal relációs algebrai műveleteket végezhetünk
 - Descartes-szorzat (\times)
 - Halmazelméleti unió (\cup)
 - Halmazelméleti különbség (\setminus)
 - Kiválasztás, szelekció (K)
 - Vetítés, projekció (V)



Alaprelációk

r_1	Vezetéknév
1	Kovács
2	Szabó

r_2	Keresztnév
1	János
2	István
3	Zoltán

Direktszorzat: $r_3 = r_1 \times r_2$

r_3	Vezetéknév	Keresztnév
1	Kovács	János
2	Kovács	István
3	Kovács	Zoltán
4	Szabó	János
5	Szabó	István
6	Szabó	Zoltán

Alaprelációk

r_1	Vezetéknév
1	Kovács
2	Szabó

r_2	Keresztnév
1	János
2	István
3	Zoltán

Unió: $r_4 = r_1 \cup r_2$

r_4	Eredmény
1	Kovács
2	Szabó
3	János
4	István
5	Zoltán

Kiválasztás:

$$r_5 = K_{Keresztnev='János'}(r_3)$$

r_3	Vezetéknév	Keresztnév
1	Kovács	János
2	Kovács	István
3	Kovács	Zoltán
4	Szabó	János
5	Szabó	István
6	Szabó	Zoltán

r_5	Vezetéknév	Keresztnév
1	Kovács	János
2	Szabó	János

Vetítés:

$$r_6 = V_{Vezeteknev}(r_5)$$

r_6	Vezetéknév
1	Kovács
2	Szabó

Relációk példa

Ugyanez SQL-ben:

```
SELECT Vezetéknév  
FROM Nevek  
WHERE Keresztnev = 'János'
```



Hogyan alakítsuk ki a relációkat, táblákat?

- Az adatbázis későbbi használhatósága attól függ, hogy mennyire alaposan terveztük meg az adatbázis szerkezetét
- Ez egy önálló tudományág (Adatbázis-tervezés), de az alapjait mindenkinek érdemes megismernie
- Alapelv: Olyan adatbázist tervezzünk, ami a rendszeres használat során is sértetlen marad!
- Mit jelent a sértetlenség?
 - Úgy kell kialakítani a táblákat, hogy új adatok beszúrása, módosítása, régiek törlése esetén is megmaradjon a logikai szerkezete
 - Az alapvető cél a redundancia csökkentése, lehetőleg teljes megszüntetése
 - Ebben az esetben redundancia alatt nem a szövegszerű ismétlődést értjük, hanem a funkcionális függőség miatt előálló ismétlődést



Outline

- 1 Adatbázis-kezelés
 - Az adatbázis fogalma
 - Tranzakciókezelés
- 2 Adatbázis-kezelő rendszerek
 - Relációs adatmodell
 - Funkcionális függőség



Redundancia az adatbázisban

Dolgozók munkahelye

r_{mh}	Dolgozó neve	Cégnév	Mh. város	Irányítószám
1	Molnár Piroska	Zalaegerszegi Kórház	Zalaegerszeg	8900
2	Horváth Katalin	Zalaegerszegi Kórház	Zalaegerszeg	8900
3	Egerszegi Judit	Zalaegerszegi Kórház	Zalaegerszeg	8900
4	Lenti Edit	Soproni Kórház	Sopron	9400
5	Keleti Edina	Soproni Kórház	Sopron	9400

- Jó ez az adatbázis?
- Mi a baj vele?
- Hogyan javítsuk ki?



Redundancia az adatbázisban

Dolgozók munkahelye

r_{mh}	Dolgozó neve	Cégnév	Mh. város	Irányítószám
1	Molnár Piroska	Zalaegerszegi Kórház	Zalaegerszeg	8900
2	Horváth Katalin	Zalaegerszegi Kórház	Zalaegerszeg	8900
3	Egerszegi Judit	Zalaegerszegi Kórház	Zalaegerszeg	8900
4	Lenti Edit	Soproni Kórház	Sopron	9400
5	Keleti Edina	Soproni Kórház	Sopron	9400

- Jó ez az adatbázis?
- Mi a baj vele?
- Hogyan javítsuk ki?



Redundancia az adatbázisban

Dolgozók munkahelye

r_{mh}	Dolgozó neve	Cégnév	Mh. város	Irányítószám
1	Molnár Piroska	Zalaegerszegi Kórház	Zalaegerszeg	8900
2	Horváth Katalin	Zalaegerszegi Kórház	Zalaegerszeg	8900
3	Egerszegi Judit	Zalaegerszegi Kórház	Zalaegerszeg	8900
4	Lenti Edit	Soproni Kórház	Sopron	9400
5	Keleti Edina	Soproni Kórház	Sopron	9400

- Jó ez az adatbázis?
- Mi a baj vele?
- Hogyan javítsuk ki?



Funkcionális függőség

- Egy A jellemző (oszlop) funkcionálisan függ egy B jellemzőtől, ha egy reláció (kitöltött táblázat) minden s és t elemére (sorára), ha s és t értéke megegyezik a B jellemzőn, akkor A értéken is megegyeznek
- Jelölés: $B \rightarrow A$
- Értelemszerűen: $A \rightarrow A$
- Nem kívánatos redundancia: a funkcionális függőség alapján levezethető információ ismételt tárolása
- Ezt kell megszüntetni!
- Miért?



Anomáliák I

Vegyük észre, az r_{mh} relációban:

- [Cégnév] \rightarrow [Mh. város]
- [Cégnév] \rightarrow [I.r. szám]
- [I.r. szám] \rightarrow [Mh. város]

1 Beszűrési anomália

Tegyük fel, hogy be akarjuk szűrni az r_{mh} táblázatba a következő sort:

r_{mh}	Dolgozó neve	Cégnév	Mh. város	I.r. szám
6	Kelemen Evelin	Zalaegerszegi Kórház	Sopron	7500



Anomáliák II

2 Törlési anomália

- Tegyük fel, hogy az r_{mh} táblázatban kitöröljük a Soproni Kórház dolgozóit 4., 5. sor
- Ekkor törlődik a Soproni Kórház címe is (nincs hol tárolni a továbbiakban)!

3 Módosítási anomália

- Tegyük fel, hogy megváltozik a Soproni Kórház neve, pl. Soproni Erzsébet Kórház
- Ekkor ezt az egyetlen változtatást a kórház minden dolgozójának sorában el kell végezni

Hogyan lehetne kiküszöbölni ezeket az anomáliákat?



Táblák dekompozíciója

- A funkcionális függőségek mentén bontsuk kisebb darabokra azokat a táblákat, amelyekben redundanciát találunk
- Ha a jellemzők (oszlopok) egy X részhalmaza a séma (tábla) minden jellemzőjét meghatározza, akkor azt mondjuk, hogy X **superkulcs**
- Ha X egyetlen jellemzőből áll, vagy nem hagyható el jellemző belőle anélkül, hogy a superkulcs tulajdonság sérülne, akkor azt mondjuk, hogy X **kulcs**
- Célszerű minden táblában egy egy tagú kulcsot definiálni, pl. egy sorszámot
- A különböző táblákat ezekkel a kulcsokkal kapcsoljuk össze
- Így tudjuk előállítani az eredeti, összetett táblát



Táblák dekompozíciója

r_{mh}	Dolgozó neve	Cégnév	Mh. város	Irányítószám
1	Molnár Piroska	Zalaegerszegi Kórház	Zalaegerszeg	8900
2	Horváth Katalin	Zalaegerszegi Kórház	Zalaegerszeg	8900
3	Egerszegi Judit	Zalaegerszegi Kórház	Zalaegerszeg	8900
4	Lenti Edit	Soproni Kórház	Sopron	9400
5	Keleti Edina	Soproni Kórház	Sopron	9400

r_{dolg}	Dolgozó neve	Cég
1	Molnár Piroska	zk
2	Horváth Katalin	zk
3	Egerszegi Judit	zk
4	Lenti Edit	sk
5	Keleti Edina	sk

r_{ceg}	Cégnév	Irsz
zk	Zalaegerszegi Kórház	8900
sk	Soproni Kórház	9400

r_{irsz}	Város
8900	Zalaegerszeg
9400	Sopron



Adatbázis-tervezés

- Gyakorlatban persze nem kell utólag feldarabolni a táblákat
- Eleve olyan táblákat hozunk létre, amelyekben nincsen redundancia
- Minden táblában csak egyféle, összetartozó adatelemeket sorolunk fel
- Az a jó, ha egy egyszerű mondattal meg tudjuk fogalmazni az egyes táblák tartalmát
 - 1 Melyik városnak mi az irányítószáma?
 - 2 Melyik kórház melyik városban található (irányítószámmal)?
 - 3 Hol dolgoznak az emberek?
- Az nem volt jó tábla, hogy
 - hol dolgoznak az emberek, melyik városban, milyen irányítószám alatt?



Adatbázis-tervezés

- Minden táblában definiálunk egy-egy kulcsot, és ezekkel hivatkozunk rá egy másik táblából
- Így több különböző táblából is használhatjuk a tábla adatait
- Például a fenti példában csak egy helyen kell nyilvántartanunk a városok irányítószámait, és bárhol felhasználhatjuk
- Ezért jelent minőségi változást az adatbázis az adattáblához képest

(Ezekben a példákban az adatbázisok szerkezetére koncentráltunk. Természetesen lehetnek más szempontok is, amelyek felülbírálnak ezeket a szempontokat. Például a program gyorsabbá tétele érdekében néha engedélyezzük a redundanciát.)



Összefoglalás

- 1 Adatbázis-kezelés
 - Az adatbázis fogalma
 - Tranzakciókezelés

- 2 Adatbázis-kezelő rendszerek
 - Relációs adatmodell
 - Funkcionális függőség



Befejezés

Köszönöm a figyelmet!

