Dr. Kotsis Domokos

Adatbázisok

Segédanyag az előadáshoz

Cél

Cél: információ tárolás, feldolgozás, továbbítás.

Az információ feldolgozás alapvető módszerei

- **Szóbeli**
- **Írásbeli**
  Nyomtatott
- **Elektronikus**

A szóbeli információ feldolgozás

Az információ tárgya akkor, ott nem kell, hogy jelen legyen.

Szóbeli leírás: magas absztrakciós szint.
Interaktív.

Mit kell tudni?

Forrás: **értelmes beszéd.**
Nyelő: **a beszéd értése.**

Kicsi különbség.

Az írásbeli információ feldolgozás

Az információ szolgáltatás tárgyán kívül annak forrása (az információ „adója”) sem kell, hogy akkor, ott jelen legyen.

Nincs metakommunkáció, nincs interakció : még magasabb absztrakciós szint.
Hang nincs, de képek, utalások lehetnek.
Nyomtatás: sokakhoz eljut.

Mit kell tudni?

Forrás: **írás tudás, eszközök használata (stilus, lúdtoll, töltőtoll, írógép).**

Nyelő: **olvasás tudás.**

Nagyobb különbség.

Az elektronikus információ feldolgozás

Az eddigieken kívül:
A korábbi módszerek határai kitágulnak (telefon, Email).
Az információ szolgáltatásnak nem egy forrása van (rádió televízió, internet).

Változatos hangok, képek (mozgók is), interaktív.
Alacsonyabb absztrakciós szint.
Mindenkihez eljuthat.

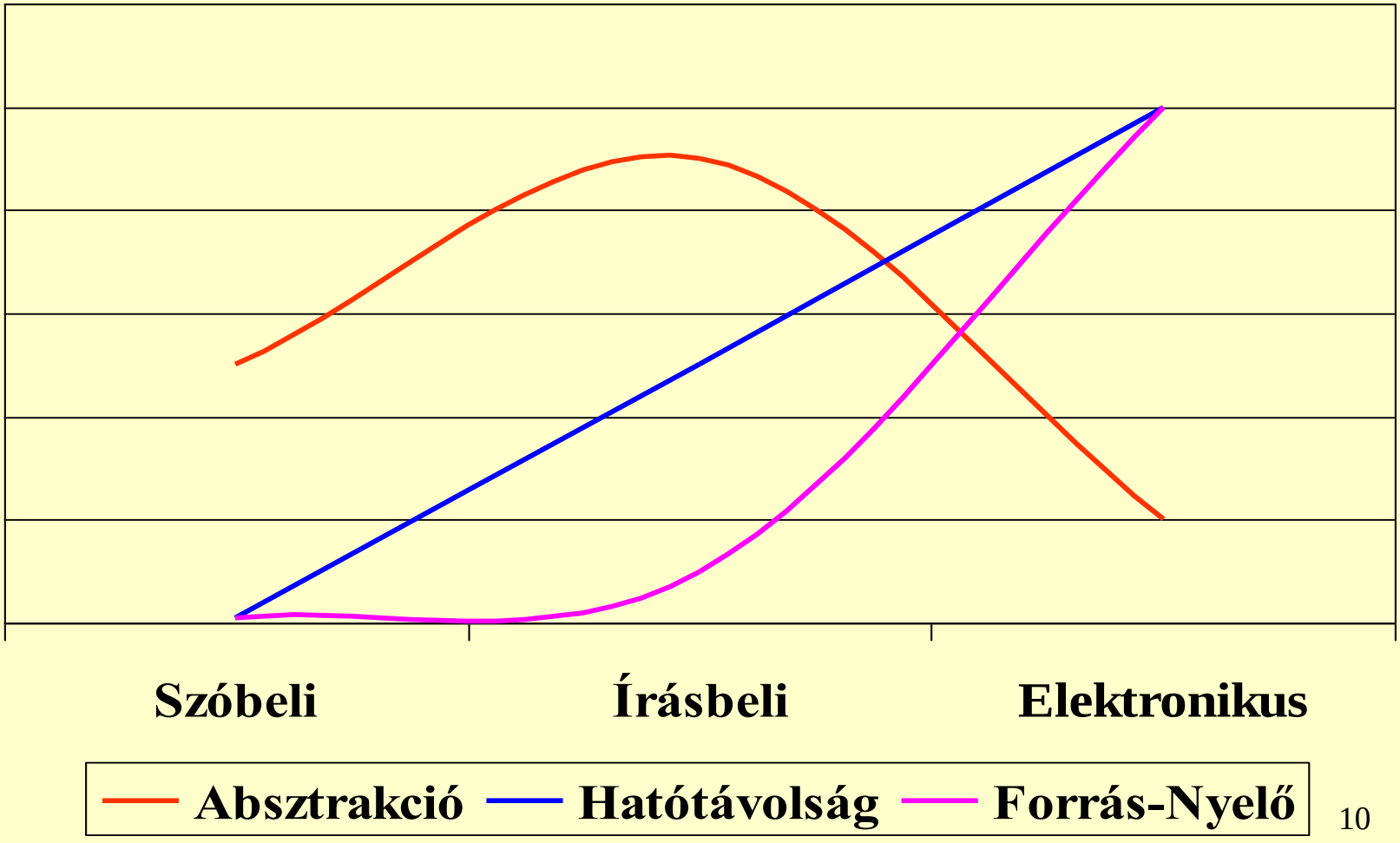
Mit kell tudni?

Forrás: **tv, rádió műsorok készítése;**
távközlési hálózat tervezése, üzemeltetése;
számítógépek hálózatok programozása.

Nyelő: **tv, rádió, telefon kezelése; számítógépek,**
programok használata (ECDL).

Óriási különbség

Összefoglalás



Szakmai bevezetés

Az információfeldolgozás kezdetei: a Hollerith gépek
A számítógépek kezdeti felhasználási területei. Számítástechnika-Informatika.
A mai helyzet: információ = energia?

Az információ feldolgozás alapvető módszerei

- **Folyamat szemléletű információ feld.**
- **Adatbázis szemléletű információ feld.**
- **Döntés-támogató rendszerek.**
- **Tudásbázisok (szakértői rendszerek).**

A folyamatszemléletű információ feldolgozás

cél → folyamat → állományok

Előnyök:

- **csak a szükséges adatokkal dolgozik**
- **optimális adatstruktúra használható**

Hátrányok:

- **többszörös tárolás (inkonzisztencia)**
- **csak egy cél**

Optimális struktúra

Opt. szemp.

- **tárolóhely**
- **feldolgozási idő**
  - **létrehozás**
  - **keresés**
  - **változtatás**
- **felhasználás egyszerűsége**

Keresés 1.

Egy N elemű file-ban egy rekord keresésének lépései:

$$S_N\ (A+B+C)$$

A keresési idő:

$$T_N = \sum_{i=1}^{N} p_i \sum_{j=1}^{S_i} (A_{i,j}+B_{i,j}+C_{i,j})$$

ahol p_i az i-edik elem keresésének valószínűsége ($\sum_{i=1}^{N} p_i = 1$),

és S_i az i-edik elem megtalálásához szükséges lépésszám.

Keresés 2.

Fontos a hardware, a tároló szerkezete!

RAM: a címek sorrendje közömbös.

Mozgófejes lemez: pozícionálás, forgás, beolvasás.

Keresés 3.

átlagos t időt véve:

$$T_N = \sum_{i=1}^{N} p_i \, t$$

egyenlő gyakorisággal számolva:

$$T_N = S_N t$$

A keresést meghatározó tényezők

Struktúra

Algoritmus

Pl. rendezetlen esetben lineáris keresés:

$S_N \approx N/2 \quad S'_N \approx N$;

ugyanez rendezett esetben:

$S_N \approx N/2 \quad S'_N \approx N/2$;

de itt nyilván van jobb módszer is.

A legfontosabb állomány struktúrák

- **Szekvenciális állomány struktúrák**
- **Hierarchikus állomány struktúrák;**
- **Hálós állomány struktúrák;**
- **Nem konzekutív állomány struktúrák**

Fizikai szekvenciális állomány struktúrák

A rekordok logikai és fizikai sorrendje megegyezik.

Keresési módok:

- **Lineáris keresés**
- **Bináris, logaritmikus keresés**
- **Peterson-féle keresés**

Bináris, logaritmikus keresés

Minden lépésben hosszra nézve felezzük a vizsgálandó állományt.

$$S_N \approx S'_N \approx \log_2(N)$$

Heurisztikus bizonyítás:

legyen K olyan, hogy $N=2^K$.

Minden felezésnél $K_{új} = K_{régi} - 1$.

$K_{új}=0$ éppen innen K lépés után lesz, és

$$K=\log_2(N)$$

Peterson-féle keresés

Minden lépésben az előfordulás valószínűségére nézve felezzük a vizsgálandó állományt.

Egyenletes eloszlás mellett:

$$A = A_E + \frac{A_U - A_E}{K_U - K_E} * K$$

ekkor:

$$S_N \approx S'_N \approx \tfrac{1}{2} \log_2(N)$$

Bináris vs. Peterson keresés

S_N a Petersonnál kisebb, de

1. az eloszlás nem feltétlenül egyenletes,

2. Petersonnál valódi osztás kell, a binárisnál a léptetés alkalmazható.

A leggyorsabb, ha $N = 2^k - 1$ alakú.

Ha nem ilyen:

1. két részre osztás

2. kiegészítés álrekordokkal

A fizikai szekvencialitás előnye, hátránya

A keresés lépésszáma kicsi, de a rendezettség biztosítása is időigényes!

1. bonyolult beszúrási algoritmus

2. rendszeres fizikai rendezés

Logikai szekvenciális állomány struktúrák

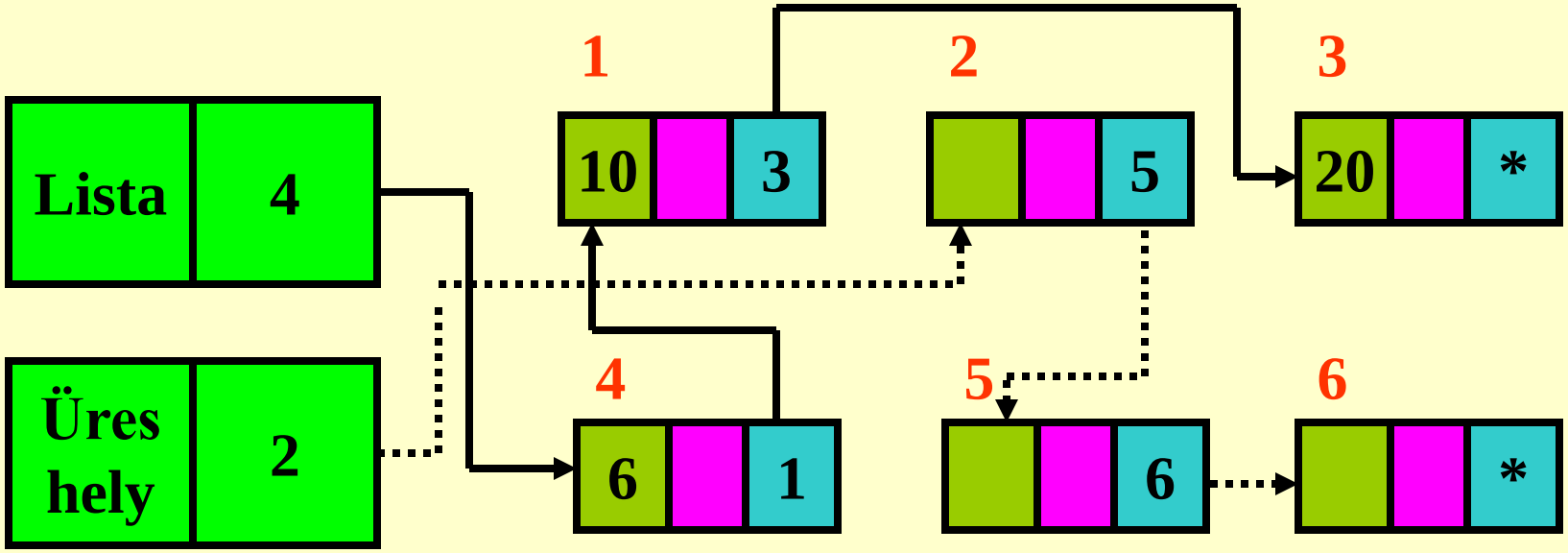
A rekordok logikai és fizikai sorrendje nem egyezik meg.

Mutatórendszerek (egy- kétirányú, gyűrűs).

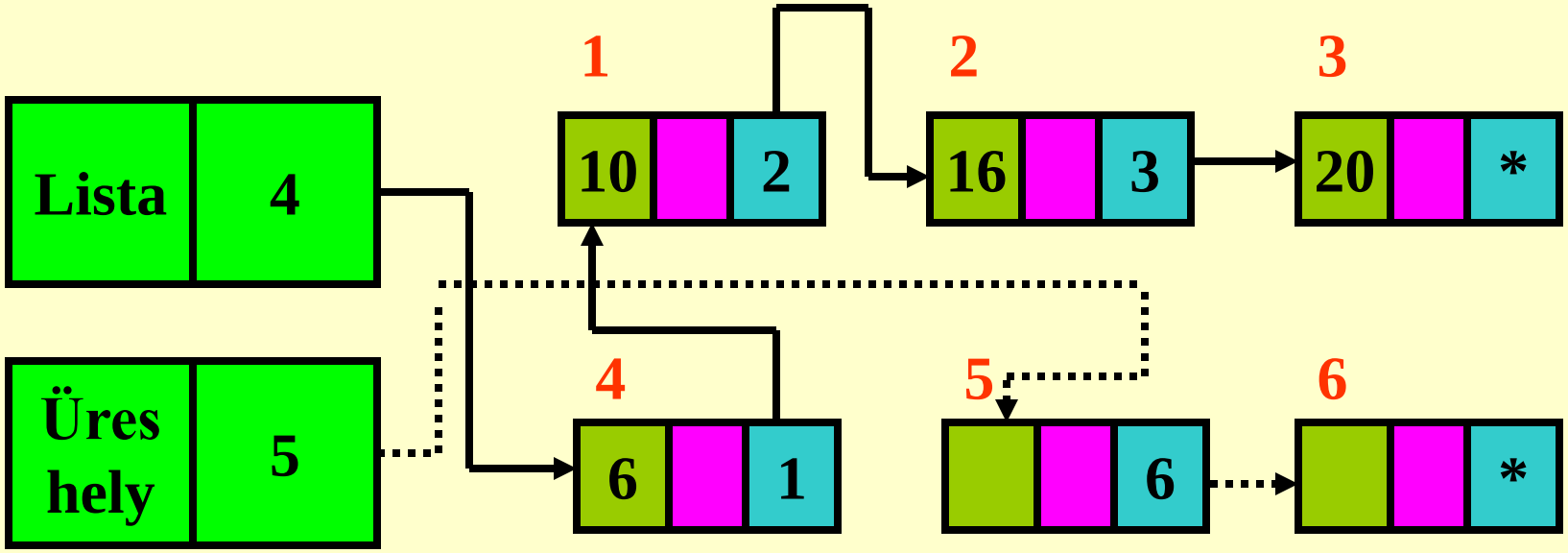
Indító tábla.

Dinamikus file-oknál jó: csak mutatókkal kell dolgozni.

Példa

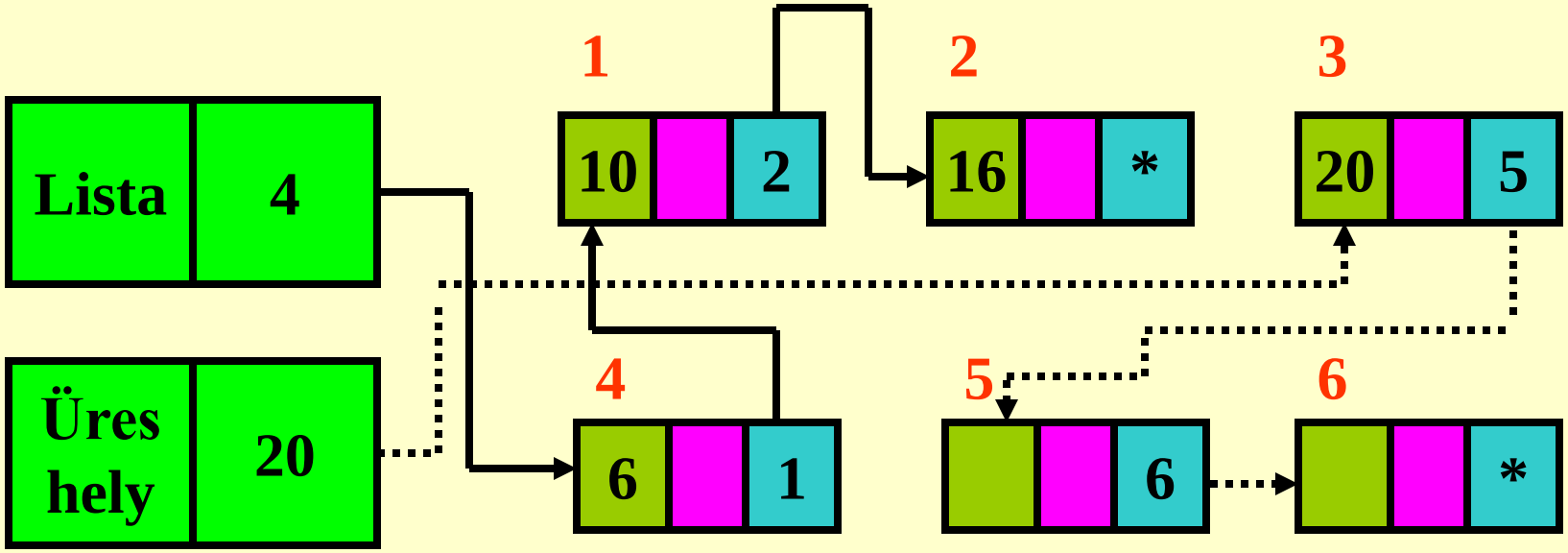


Beszúrás



Új elem: 16

Törlés



Törlendő: 20

Hátrányok

A fizikainál ismert keresési algoritmusok nem használhatók.

Mozgófejes lemeznél:

Pozícionálás, forgási idő kell az eléréshez.

Véletlenszerű elhelyezkedésnél C cilindernyi anyagnál átlagosan C/3 cilindernyi pozícionálás kell.

Csaknem fizikai szekvenciális file

Létrehozáskor fizikai szekvenciálisan elhelyezett logikai szekvenciális file.

Feldolgozáskor logikai szekvenciális.

Túlcsordulási terület a cilinderen.

Kupacos keresés 1.

N rekord, G rekord/ csoport, N/G csoport.

$$\sum_{k=1}^{\frac{N}{G}} k\left(\frac{1}{\frac{N}{G}}\right) = \frac{\frac{N}{G}+1}{2}$$

Kupacos keresés 2.

$$\overline{S}_N \approx \frac{\frac{N}{G}+1}{2} + \frac{G-1}{2} = \frac{1}{2}\frac{N}{G} + \frac{G}{2}$$

Kupacos keresés 3.

A minimum kereséshez G szerint deriválva:

$$\frac{N}{G^2} = \frac{1}{2}$$

Ennek minimum helye: $G = \sqrt{N}$

Ebből következően: $\overline{S}_N \approx \sqrt{N}$

Gyakoriság szerint rendezett file

Statikus, dinamikus megoldás.

Önrendező algoritmusok.

Hierarchikus struktúrák 1.

Egy megelőző, több rákövetkező (fa).

Nyelvi leírás: COBOL, LISP

Hierarchikus struktúrák 2.

Ábrázolás:

A.) Belső mutatós módszerek

1. Left-list

2. Több pointer

3. Segédrekordok

4. Gyűrűk

Hierarchikus struktúrák 3.

B.) Külső mutatós módszerek

1. Táblázatok

2. Bináris mátrixok

Hálós struktúrák 1.

1. Visszavezetés hierarchikusra.
2. Az ott leírt módszerek?
    A.) Belső mutatós módszerek
        1. Left-list **nem**
        2. Több pointer **igen**
        3. Segédrekordok **nem**
        4. Gyűrűk **nem**

Hálós struktúrák 2.

B.) Külső mutatós módszerek

1. Táblázatok igen

2. Bináris mátrixok igen

Nem konzekutív struktúrák

Indexelt szervezés

Direkt szervezés

Indexelt szervezés

1. Sűrű indexelés

2. Ritka indexelés

Sűrű indexelés

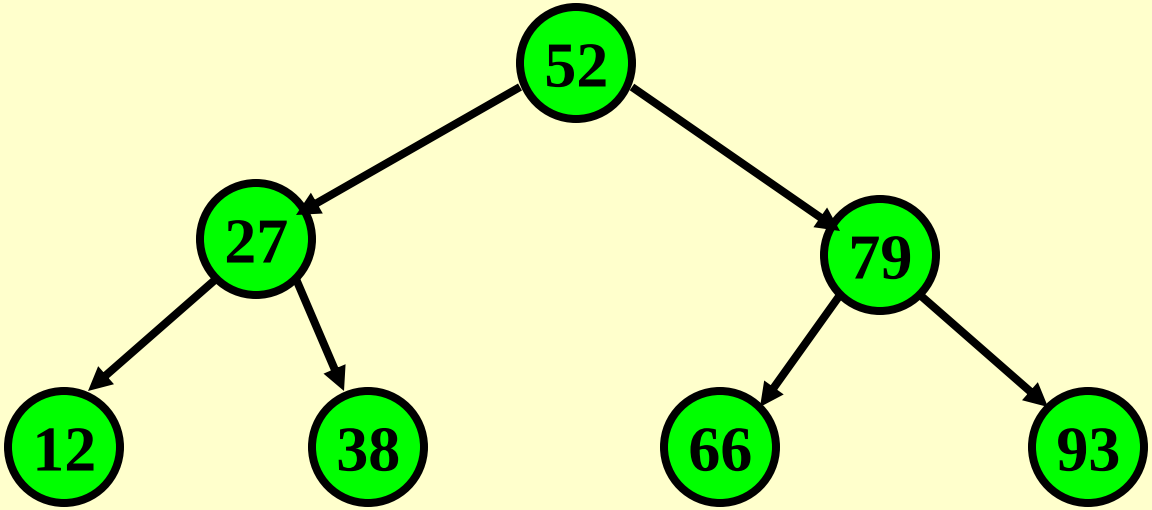
Minden rekordhoz tartozik index bejegyzés

Előnyök: több index szervezhető

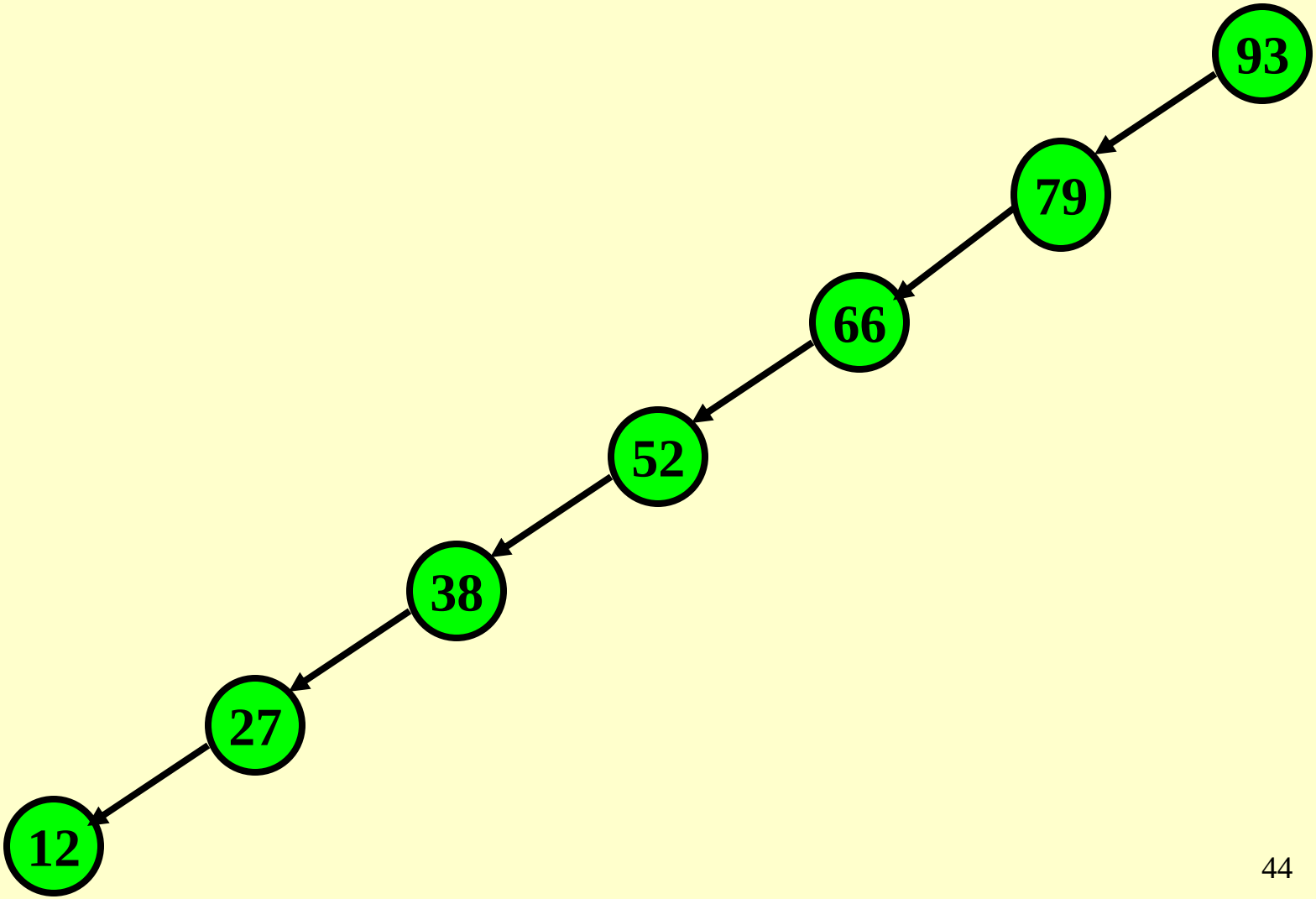
Hátrányok: nagyok az index file-ok

Mikor lehet gyors a keresés?

Bináris fa I.



Bináris fa II.



B fák

R. Bayer 1970

n a B fa rendje

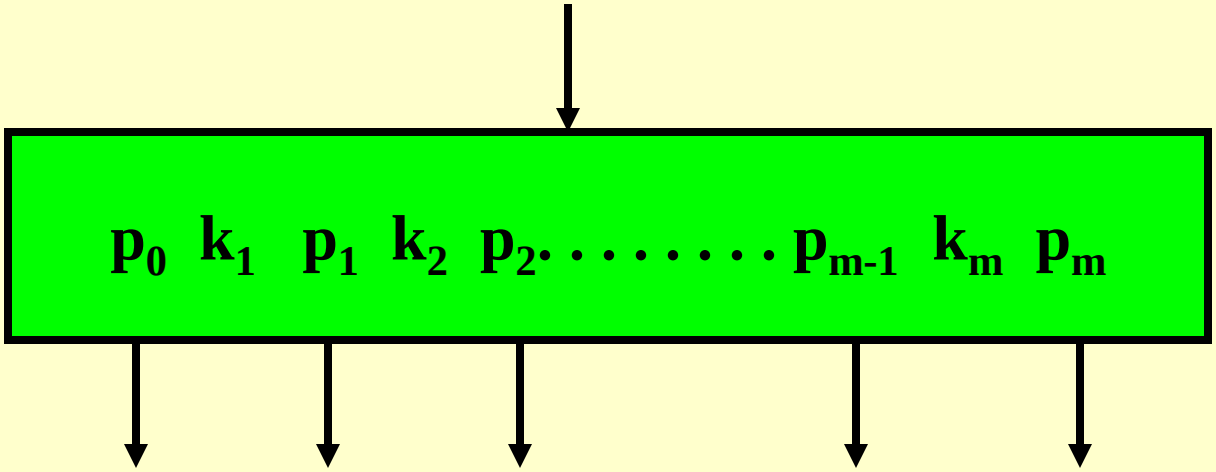
1. **Minden lap legfeljebb 2n tételt tartalmaz.**
2. **Minden lapon – a gyökér kivételével - legalább n tétel van .**
3. **Ha egy lapon m tétel van, vagy levéllap, vagy m+1 utódja van.**
4. **Minden levéllap ugyanazon a szinten van.**

B fák elérése

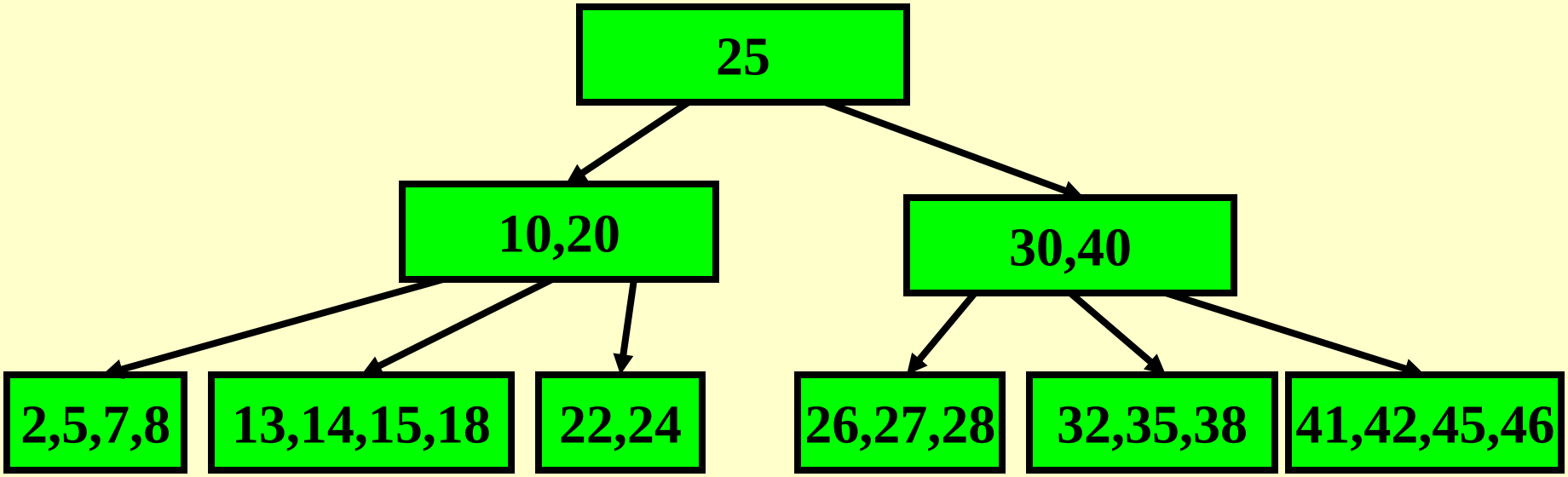
A kereséshez szükséges átlagos lépésszám a laphivatkozásoktól függ, ezek száma N elemnél n rendű fában:

$$\log_n N$$

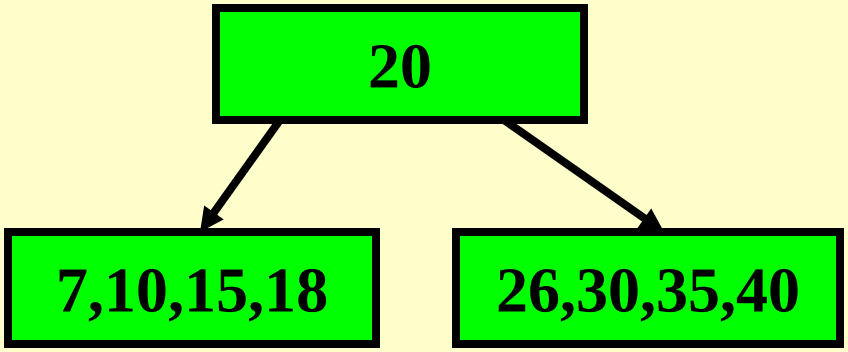
B fa egy lapja

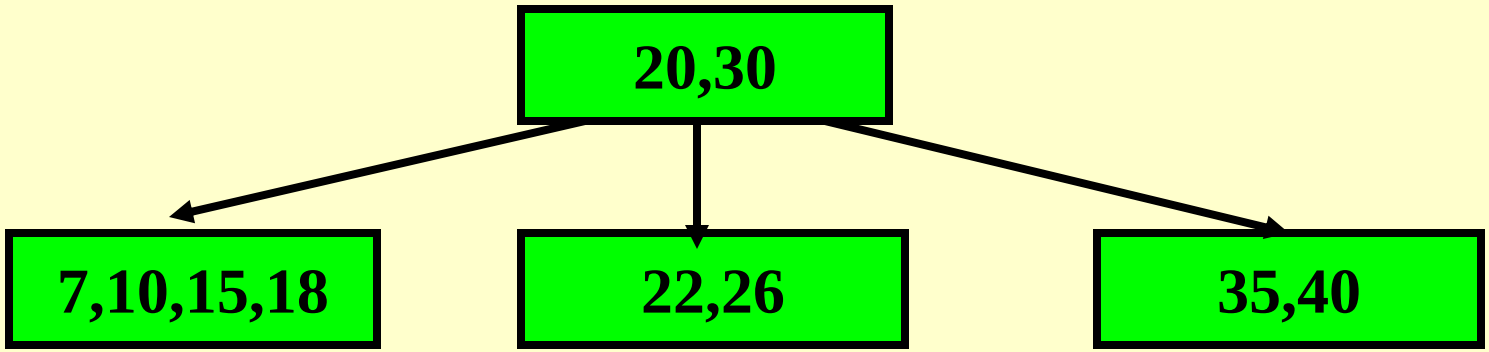


Másodrendű B fa

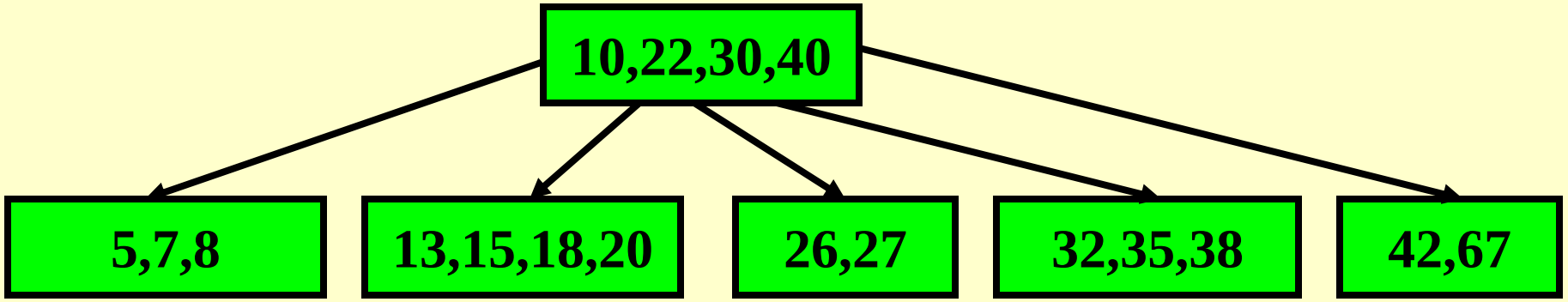


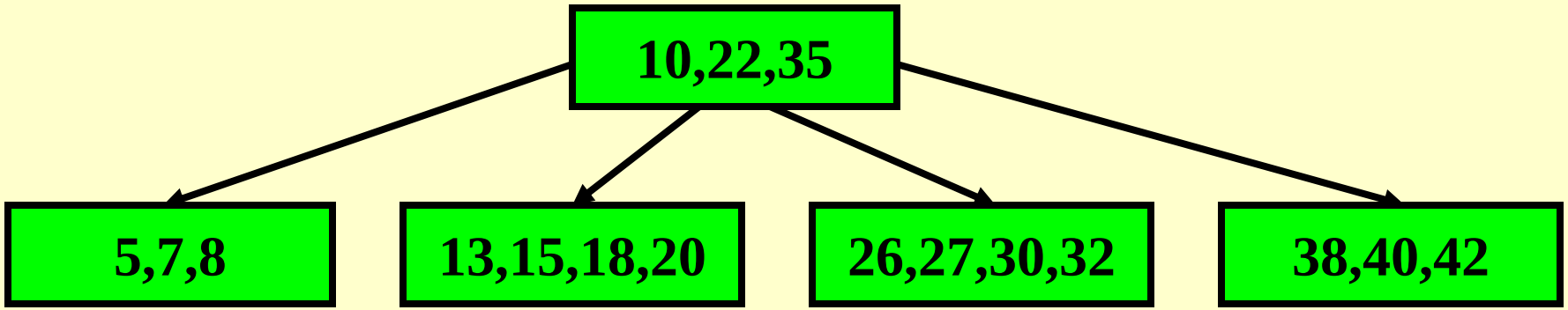
Egy új (22) kulcs beszúrása





Egy kulcs (67) törlése





B+ fák

A gyakorlatban index állományok készítésére az u.n. B+ fákat használják. (Ilyen struktúrát alkalmaz – sok más rendszerrel együtt – az Oracle is.)

A B+ fa egy olyan B fa, melyben azok a pointerek, melyek adatokra (adatok címére) mutatnak, s amelyeket keresünk, valamennyien a levéllapokon helyezkednek el, így a levéllapok és a többi lap struktúrája különbözik.

Ritka indexelés

„Jelző cölöpök” mutatják a rekord helyét.

Rendezettség kell!

Több szintű index rendszer is készíthető.

Gyors elérés, de csak egy szempont szerint.

Index-szekvenciális szervezés 1.

A ritka indexelés egy megvalósítása.

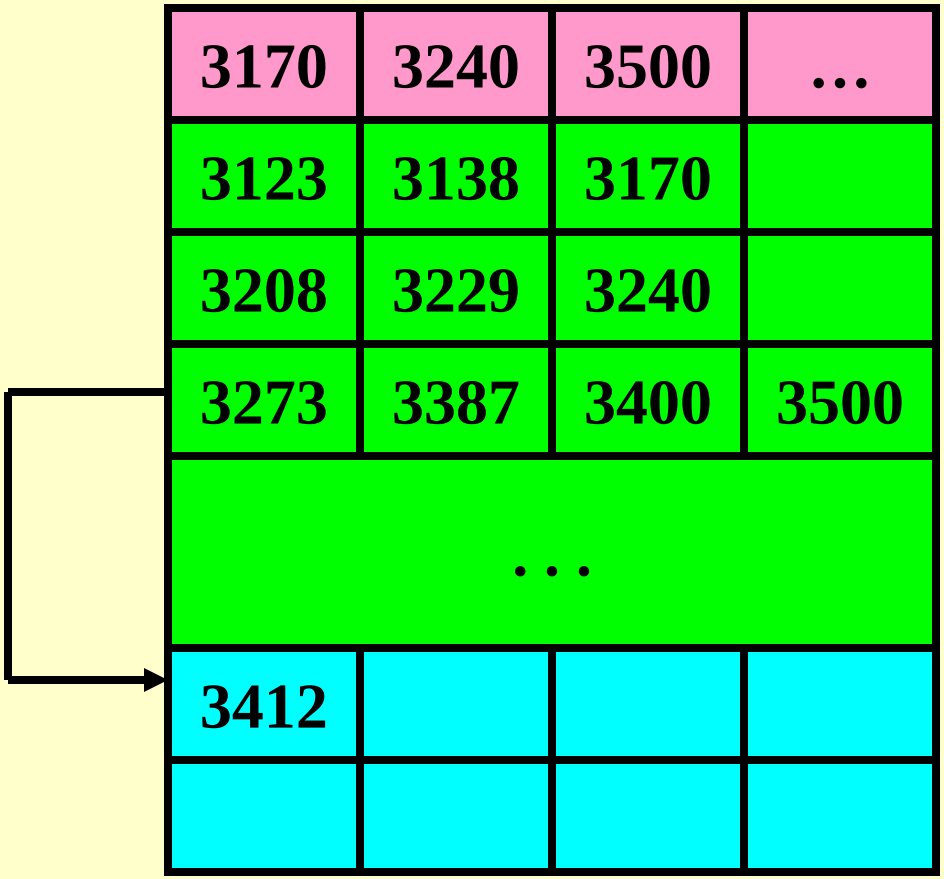
A fizikai architektúrának megfelelően:

Sáv; cilinder; fő index.

Index; elsődleges; túlcsordulási terület.

Szervezés: az elsődleges területen fizikai, a túlcsordulási területen logikai szekvenciális szervezés.

A tároló felosztása

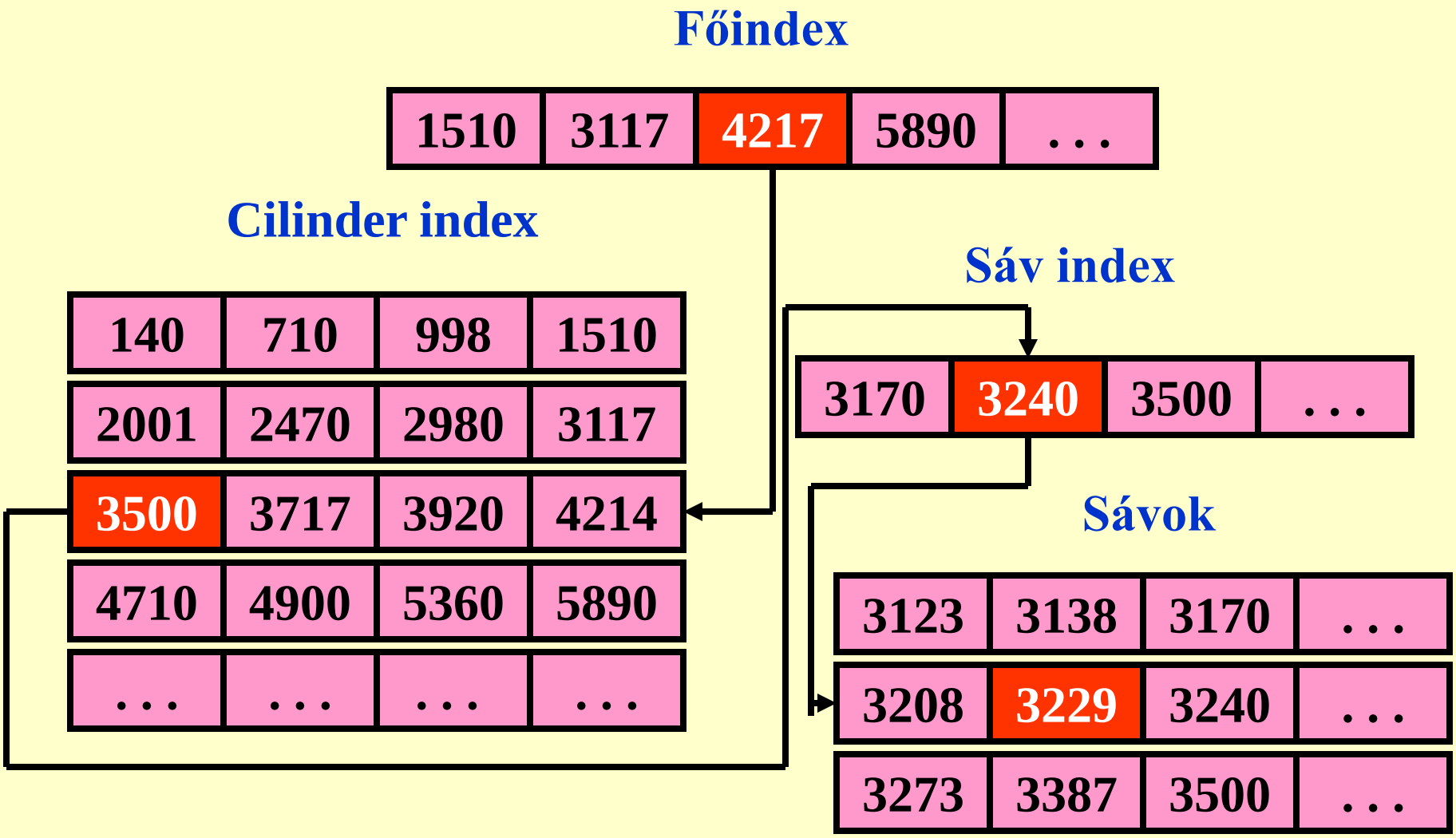


Sávindex

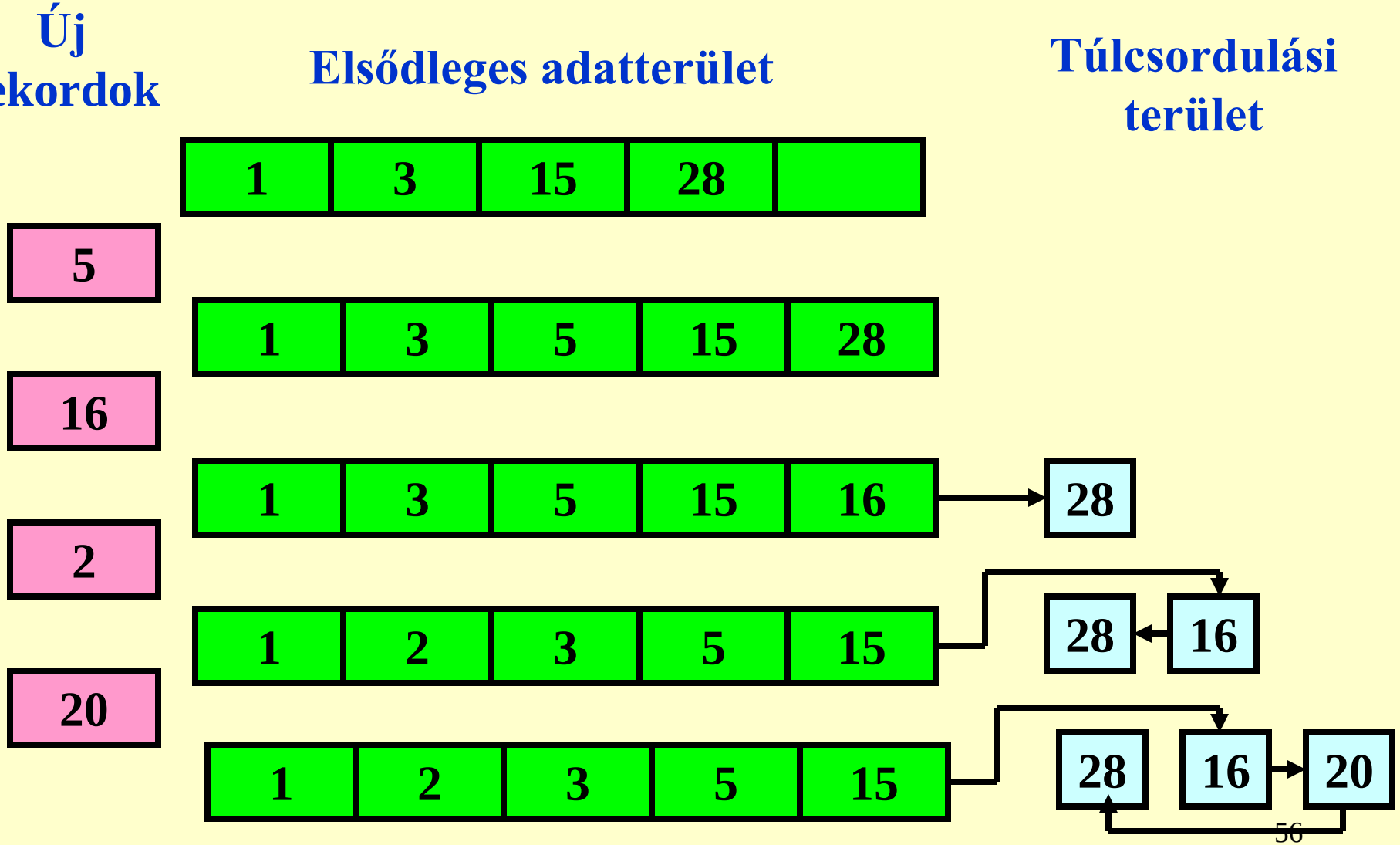
Elsődleges terület

Túlcsordulási terület

Egy rekord (3229) megkeresése



Rekordok beszúrása



Direkt szervezés 1.

Egy leképezést adunk meg a rekord kulcsa (K) és a címe (A) között.

$$A = f(K)$$

Kívánatos lenne az egy-egy értelműség:

$$K_1 \neq K_2 \Rightarrow f(K_1) \neq f(K_2)$$

Direkt szervezés 2.

Közvetlen leképezés: igen rossz a tárkihasználás.

Hashing algoritmusok: megengedjük, hogy

$$K_1 \neq K_2$$

esetén is – lehetőleg ritkán - előfordulhasson

$$f(K_1) = f(K_2)$$

Két hashing algoritmus

Maradék módszer: M modulus mellett

$$A = R\left(\frac{K}{M}\right)$$

Csonkítás: a kevés információt hordozó jegyek elhagyása.

Szinonímok

A hashing algoritmusok nem egy-egy értelműek, így szinonimok keletkezhetnek.

Szinonim kezelő módszerek:

Külső láncolás

Belső láncolás

Nyílt (Peterson) módszer

Többszörös hashing

Bucket-ek használata

A külső láncolás

A szinonimokat mutatók kötik össze.

Külön túlcsordulási terület.

Problémák:

- **Tárkihasználás**
- **Törlés: az elsőt nem lehet törölni.**

A belső láncolás

A szinonimokat itt is mutatók kötik össze, de azok az (egyetlen) elsődleges terület még nem használt helyeire kerülnek, az eredeti címhez a lehető legközelebb.

A tárkihasználás jó, de másodlagos szinonimok keletkezhetnek.

A nyílt módszer

A rekordok ugyanúgy helyezkednek el, mint a belső láncolásnál, de nincsenek mutatók.

A keresésnél szükséges lépésszám nagyobb, mint a belső láncolásnál.

A többszörös hashing

Ha a leképezés szinonimra vezet, egy másikkal próbálkozunk.

(A gyakorlatban két hashing algoritmust, majd a korábban ismertetett módszerek valamelyikét használják.)

Bucket-ek használata

E módszernél nem egy-egy rekordnak, hanem egy-egy rekord csoportnak van külön címe.

Kevesebb szinonimkezelésre van szükség, de a csoportokon belül is kell keresnünk.

Hasznos akkor lehet, ha hardware szempontok (szektor) indokolják.

Példa szinonímkezelésre

35, 46, 18, 14, 63, 06,75, 85,91, 52.

Leképezés: a 11-el osztás maradéka.

Második: a 7-el osztás maradéka.

Példa bucket használatra

21, 32, 71, 14, 83, 72, 43, 51, 02, 11, 93, 44.

Leképezés: 4 bucket-be a második jegy szerint.

Összehasonlítás 1.

Az egyes módszerekhez átlagosan szükséges lépésszámot a tényleges (N), és az egyáltalán elhelyezhető (M) rekordszámmal meghatározott A=N/M u.n. kitöltési tényező függvényében vizsgáljuk.

Összehasonlítás 2.

Belső láncolásnál sikeres esetben:

$$S_N = 1 + \frac{1}{8A}\left(e^{2A} - 1 - 2A\right) + \frac{1}{4}A$$

Sikertelen esetben:

$$S_N^{'} = 1 + \frac{1}{4}\left(e^{2A} - 1 - 2A\right)$$

Összehasonlítás 3.

Nyílt módszernél sikeres esetben:

$$S_N = \frac{1}{2}\left(1 + \frac{1}{1-A}\right)$$

Sikertelen esetben:

$$S_N' = \frac{1}{2}\left[1 + \left(\frac{1}{1-A}\right)^2\right]$$

Összehasonlítás 4.

Többszörös hashing-nél sikeres esetben:

$$S_N = -A^{-1} \ln(1 - A)$$

Sikertelen esetben:

$$S_N' = (1 - A)^{-1}$$

Összehasonlítás 5.

Példaképpen sikeres esetben:

$$1+\frac{1}{8A}\left(e^{2A}-1-2A\right)+\frac{1}{4}A=1+\frac{1}{2}A+\sum_{i=2}^{\infty}\frac{2^{i-2}A^{i}}{(i+1)!}$$

$$\frac{1}{2}\left(1+\frac{1}{1-A}\right)=1+\frac{1}{2}A+\frac{1}{2}A^{2}+\ldots=1+\frac{1}{2}\sum_{i=1}^{\infty}A^{i}$$

$$-A^{-1}\ln(1-A)=1+\frac{A}{2}+\frac{A^{2}}{3}+\ldots=\sum_{i=0}^{\infty}\frac{A^{i}}{i+1}$$

Összehasonlítás 6.

Sikertelen esetben:

$$1+\frac{1}{4}\left(e^{2A}-1-2A\right)=1+\frac{1}{4}\sum_{i=2}^{\infty}\frac{(2A)^i}{i!}$$

$$\frac{1}{2}\left[1+\left(\frac{1}{1-A}\right)^2\right]=1+A+\frac{3}{2}A^2+\ldots=1+\sum_{i=1}^{\infty}\frac{i+1}{2}A^i$$

$$(1-A)^{-1}=1+A+A^2+\ldots=\sum_{i=0}^{\infty}A^i$$

Összehasonlítás 7.

Nem elég a lépésszámot vizsgálni, hisz az egyes lépések időigénye módszerenként más.

Figyelembe kell venni a hardware adottságokat is.

Összehasonlítás 8.

Gyors véletlen elérésű (pl. RAM) tárolónál:

TH: nagy algoritmusidő.

NyM: több lépés mint BL-nél.

BL: másodlagos szinonimok miatt több lépés, mint KL-nél.

KL: leggyorsabb, de rossz tárkihasználás.

Összehasonlítás 9.

Mozgófejes tárolónál:

TH, KL: sok fejmozgás.

NyM: több lépés mint BL-nél.

De NyM-hez nem kell CPU, így a leggyorsabb lehet.

Alapelv

A rendelkezésre álló adatokból indulunk ki, az „összes” adatot (entity) és a közöttük fennálló „összes” kapcsolatot (relationship) egy integrált adatbázisba gyűjtjük, és valamennyi felhasználó valamennyi kérdéséhez ezt az adatbázist (vagy ennek egy részét) használhatja.

Adatreprezentáció

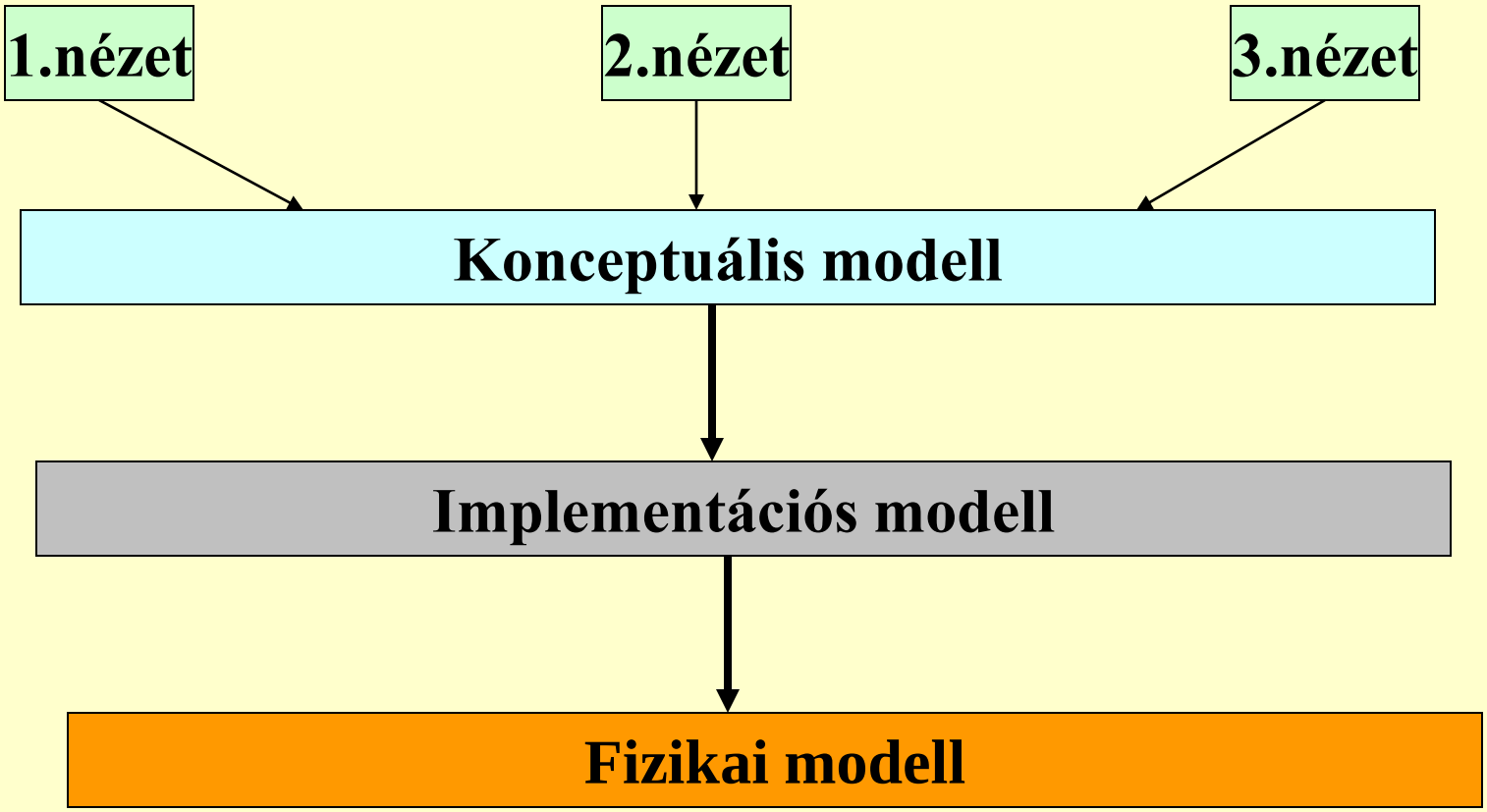
Az „összes adat” egy „mini világra” vonatkozik.

Ennek leírása több szintű:

- ***konceptuális*** **(magas szintű),**
- ***implementációs*** **(reprezentációs),**
- ***fizikai*** **modell.**

Az egyes felhasználói *nézetek* (view-k) az adatbázisnak csak egy részét látják.

A leképezések (mapping-ek)



Egyed-kapcsolat leírások

Az alapvetően használt modell az *egyed-kapcsolat* (entity-relationship), azaz ER modell.
Az egyedeknek *tulajdonságaik* (attribute) vannak, ezek egyike, vagy ezekből egy készlet az egyedet egyértelműen azonosító *kulcs*.
A kapcsolatok lehetnek 1:1, 1:n, m:n jellegűek.
A kapcsolatokat sokszor egyedhalmazzá alakítjuk (kapcsoló rekord: a kapcsolat mint egyed).

Adatbázis felügyelő

Több felhasználó: védeni kell tőlük a rendszert, és őket egymástól.
Ez az *adatbázis felügyelő* (data base administrator) egyik feladata.

Tervezés: *séma, alséma* készítés.
Eszköz: DDL.

Adatfüggetlenség

***Logikai*: az adatok logikai struktúrájában (konceptuális, implementációs modell) történő változtatás ne érintse az ezen változtatást nem igénylő felhasználók munkáját.**

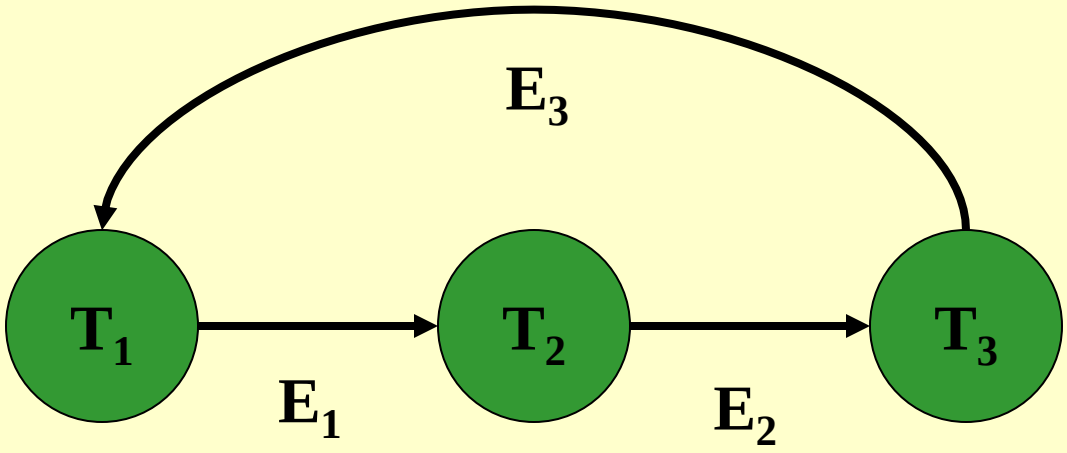
***Fizikai*: a fizikai modell változtatása ne kényszerítse ki az implementációs modell változtatását.**

A kettő együtt az *adatfüggetlenség*.

Tervezési szempontok

Kapcsolat múlttal, jövővel.
Biztonság, titkosság, pontosság.
Konkurrens folyamatok kezelése.
Válaszidő.
Kényelmes használat.
Költségek.

Várakozási gráf



Az ABF üzemeltetési feladatai

Mentések, karbantartás.

Kapcsolattartás a felhasználóval.

A DML

A felhasználó eszköze.

Többféle felosztás:

1. **Alkalmazott nyelv szerint:**
   - **Host language**
   - **Self Contained Language**
2. **A felhasználás jellege szerint:**
   - **Procedurális**
   - **Deklaratív**

Host Language (Beépített, beágyazott nyelvű) rendszerek

Egy korábban ismert magas szintű nyelvet használnak.

Megoldások:

Eljárás gyűjtemények könyvtárban.

Eljárás gyűjtemény előfordítóval.

Eljárás gyűjtemény interpreterrel.

Self Contained (Önálló) nyelvű rendszerek

Lehet külön programozható nyelv.

Lehet, csak paraméterezendő rendszer.

Lehet 4GL fejlesztő rendszer.

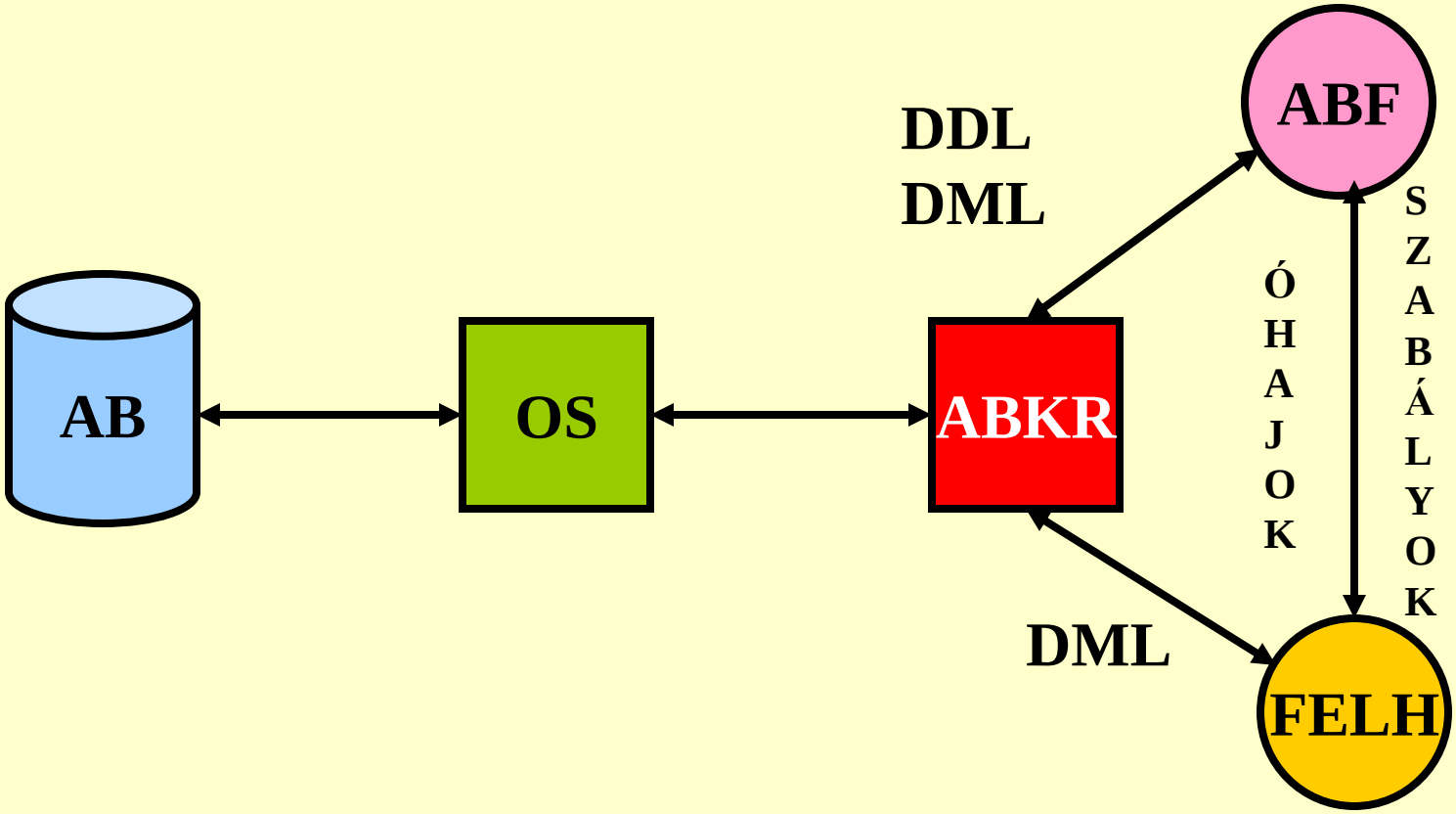
Procedurális rendszerek

Egy lépésben egy rekordot dolgoznak fel (record-in-time feldolgozás).

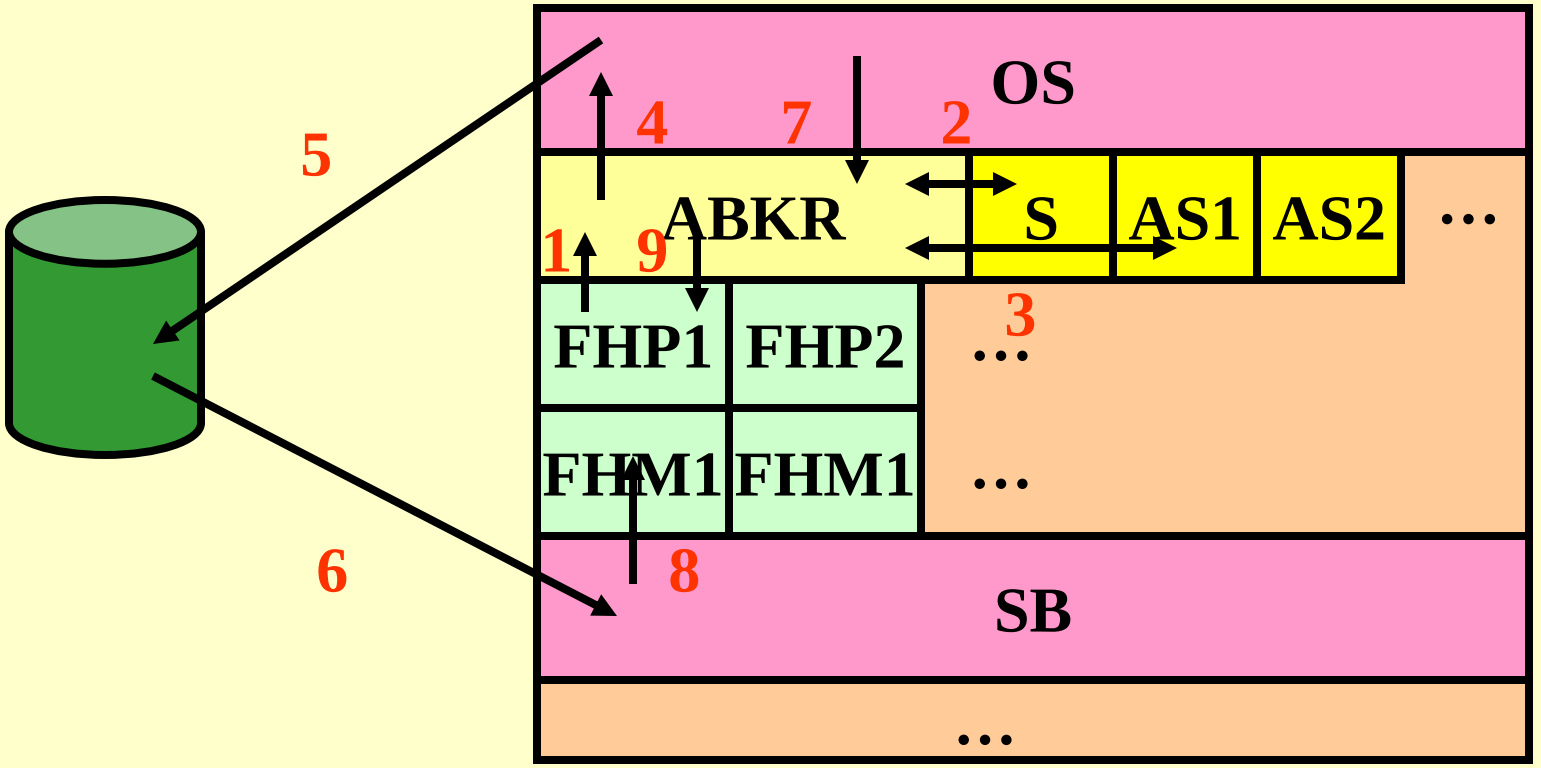
Deklaratív rendszerek

Egy lépésben egy meghatározott tulajdonságú halmazt (pl. egy táblázatot) dolgoznak fel(set-in-time rendszerek: ilyen pl. az SQL).

Az OS és az ABKR



Egy lekérés folyamata



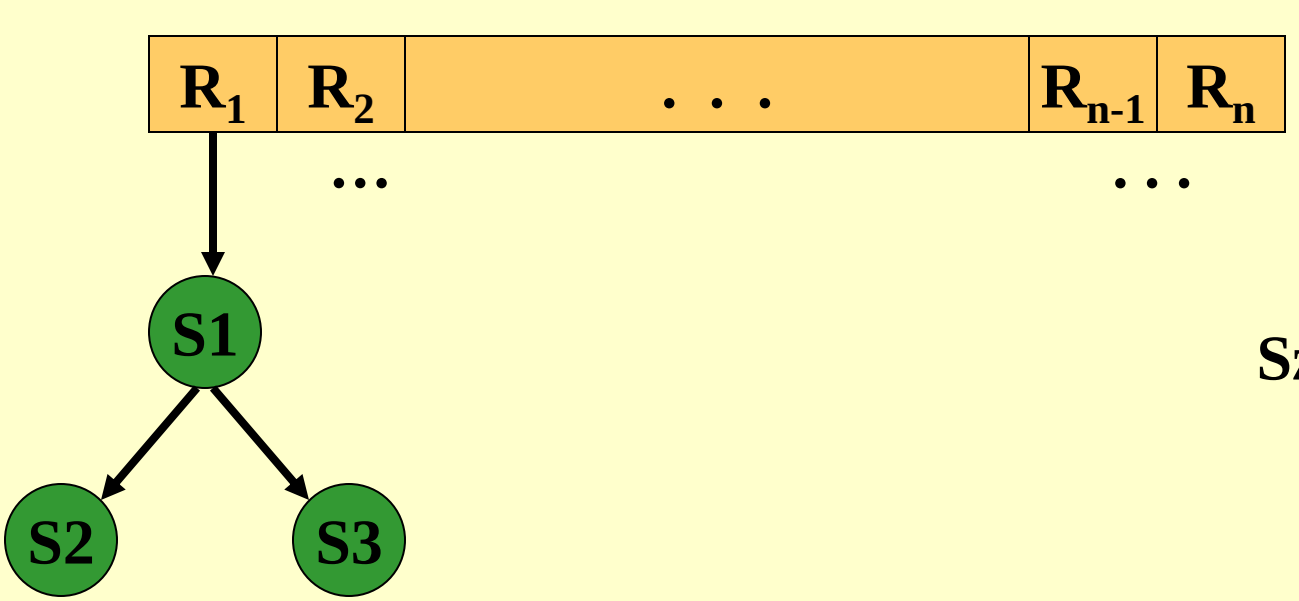
Alapvető ABKR modellek (approach)

1. **Hierarchikus**
2. **Hálós (CODASYL, DBTG)**
3. **Relációs**

A hierarchikus modell I.

A legelső modell, ma már nem használják.
Az adatokat fákban tároljuk, ahol a fák egy-egy szögpontja a *szegmens* (segment) adatokat és a további szegmensekre utaló mutatókat tartalmaz. A fák gyökérelemei hagyományos állományokba vannak szervezve. Az egyes nézetek a számukra *érzékeny szegmenseket* (sensitive segment) látják.
Jellegzetes képviselője az IMS (IBM).

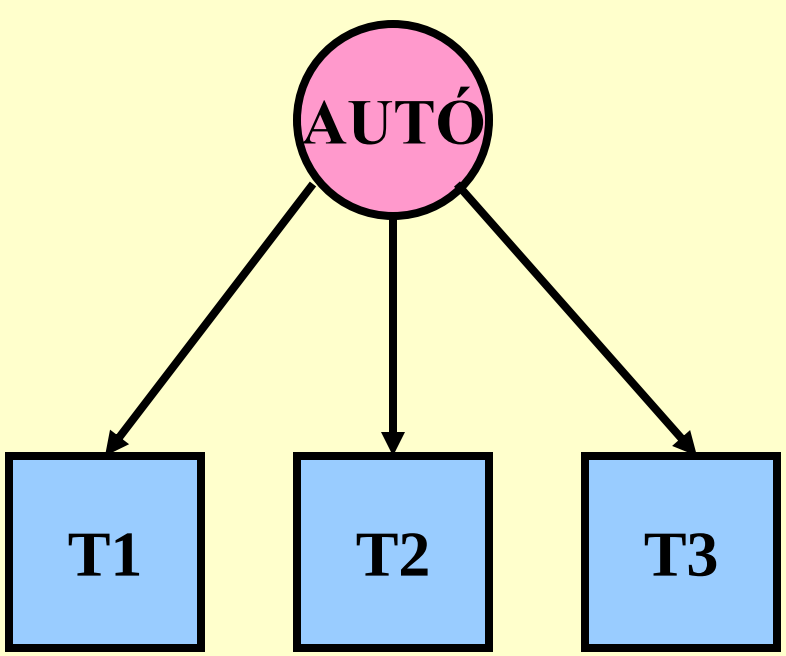
A hierarchikus modell II. (IMS)

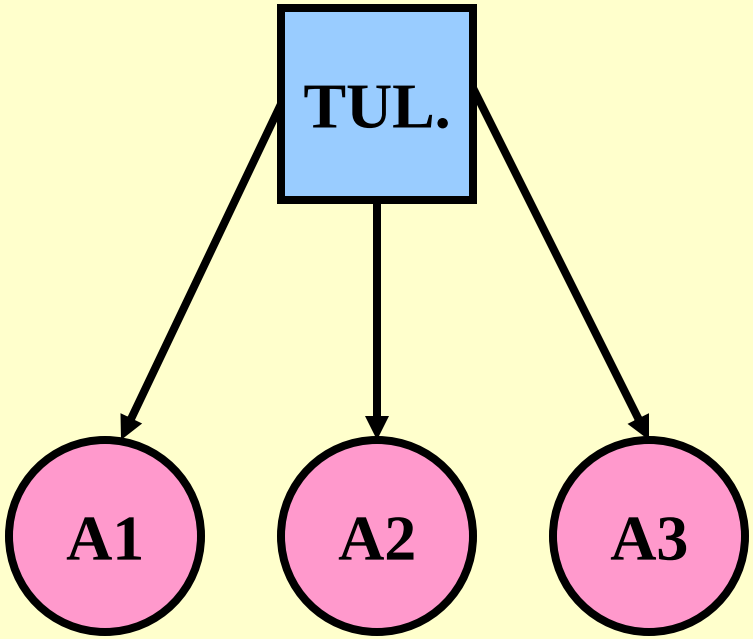


A hierarchikus modell tulajdonságai

Igazi ABKR, hisz többfelhasználós, az egyes felhasználók nem ugyanazt látják.
Nem igazi ABKR, mert a lekérdezés hatékonysága erősen függ az adatstruktúrától.

Lehetséges struktúrák egy hierarchikus modellben





A hálós modell 1.

A CODASYL bizottság által létrehozott DBTG (Data Base Task Group) jelentései (1969-1971) hozták létre.
Két évtizedig szinte kizárólag ezt használták.
Terminológiai és metodológiai javaslatokat tartalmaz.

A hálós modell 2.

Már bevezetett fogalmak:

DDL

DML

séma

alséma.

A hálós modell 3.

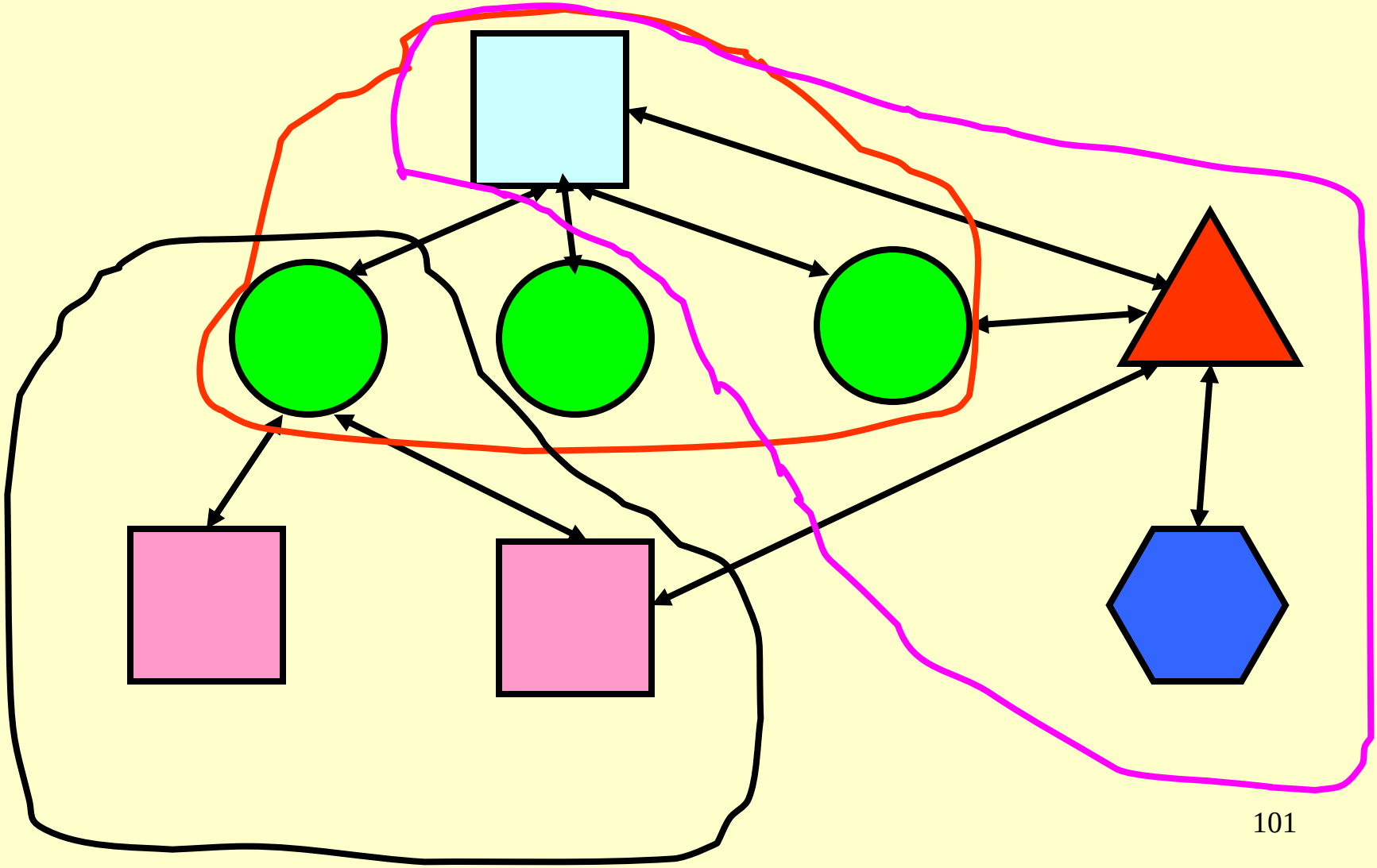
Új fogalmak:

Area: **valamilyen szempontból egységesen kezelendő adathalmaz.**

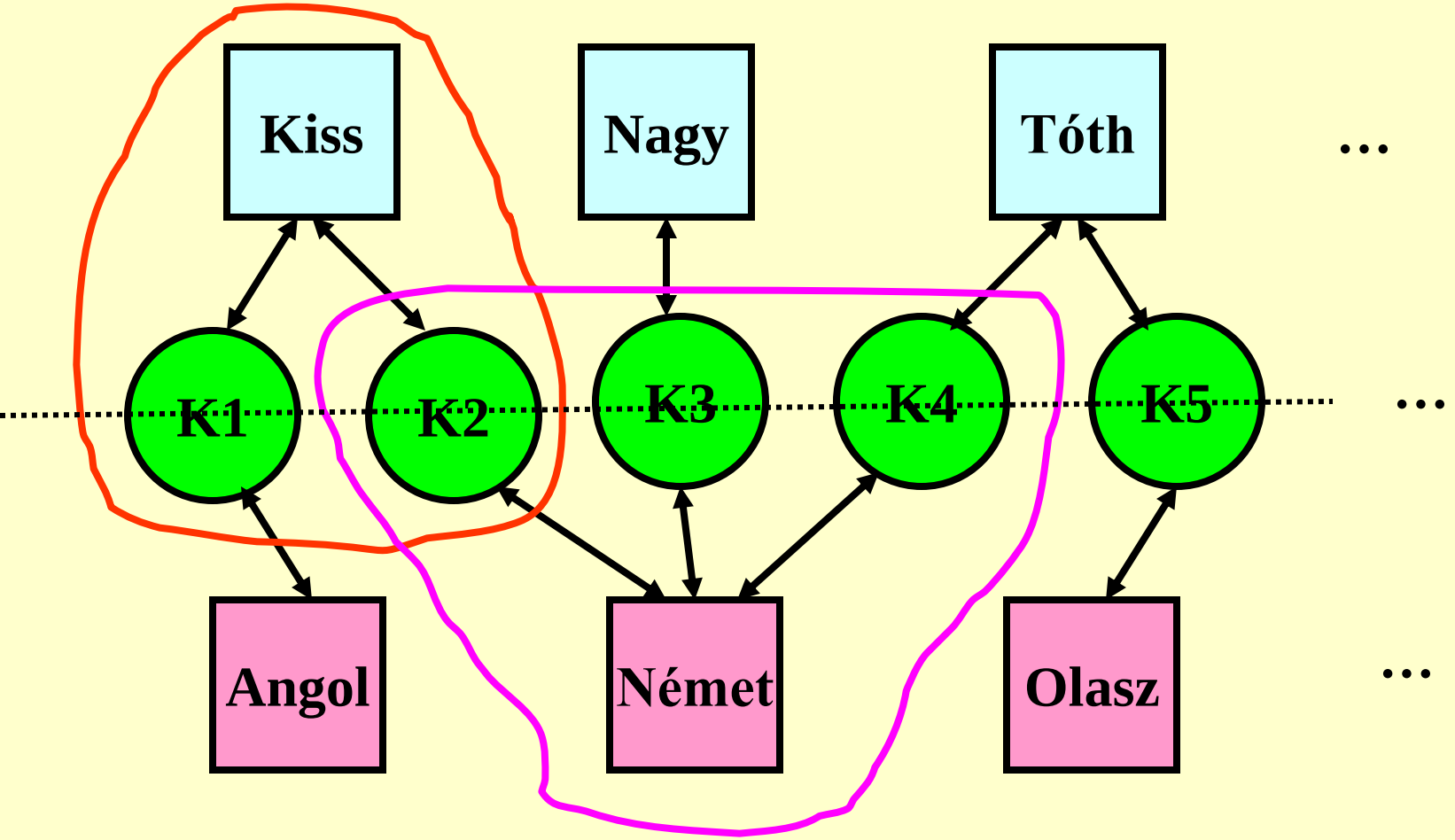
Set: **kétszintű fa, melynek gyökéreleme a *tulajdonos* (owner), levelei a *tagok* (members).**

A set-ek segítségével a legbonyolultabb hálós kapcsolatok is leírhatók.

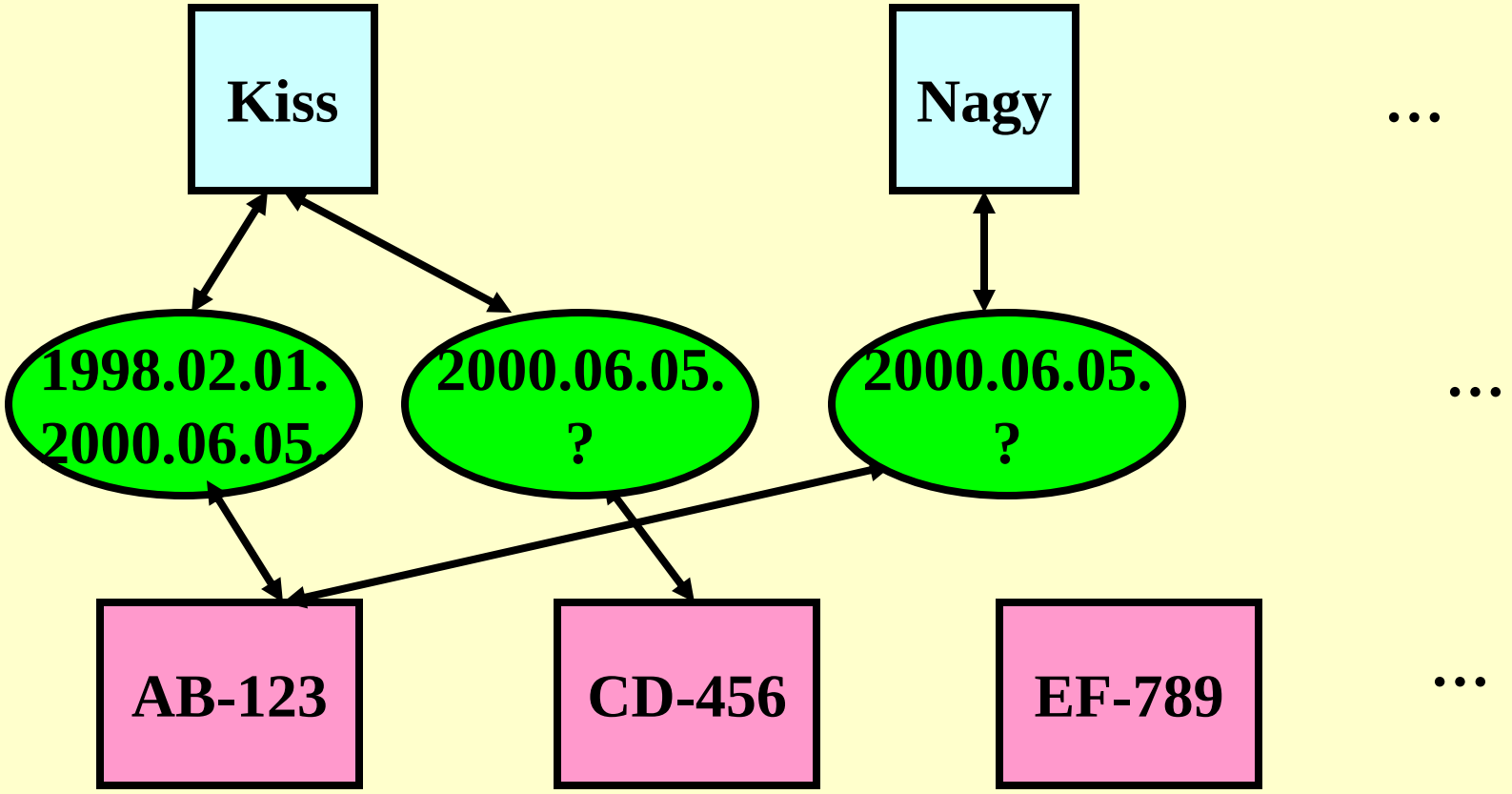
Példa SET-ekre



Kapcsolatok hálós modellben 1.



Kapcsolatok hálós modellben 2.



Séma hálós modellben

Séma név.
Zárak, kulcsok.
Rekord leírások (kalkulált mezők, kódolt mezők).
Kapcsolat leírások: SET-ek.
AREA leírások.

Alséma hálós modellben

A séma leírás valódi része.
Újdonság nem szerepelhet, de mindent át lehet nevezni.

Lekérdezés hálós modellben

Az eredeti javaslat a COBOL nyelvet javasolta (BATCH feldolgozás!), ezt váltotta fel a PL1, majd az interaktív felhasználás.

Fontos reprezentáns: IDMS (IBM 360 és 370: ESzR 1. és 2.).

Az IDMS specialitásai

Member-ek rendezése.
Indexek készítése.
Hashing algoritmus haználata.

A relációs modell 1.

A relációs modell ötlete Codd 1970-es cikkéből származik, így lényegében egyidős a DBTG jelentéssel.
Két évtized után átvette a vezető szerepet, ma szint kizárólag ezt használják, ezért ezt vizsgáljuk részletesen.

A relációs modell 2.

A *reláció* ebben az értelemben tulajdonképpen egy *táblázat* (table).
A táblázat oszlopai a *tulajdonság*ok (domains), a sorai az *n-es*ek (tuples).
Gyakran nevezik azonban a sorokat rekordnak, az egyes oszlopokhoz tartozó értékeket mezőnek (field).
Az R reláció sorainak hosszát r-el jelöljük.

Alapfeltételek relációkkal kapcsolatban

1. **Ne legyenek teljesen megegyező tartalmú sorok vagy oszlopok,**
2. **a sorok és az oszlopok sorrendje ne hordozzon információt.**

Azt a mezőt, vagy mezőkészletet, mely a sort (a sor többi elemét) egyértelműen meghatározza, *szuperkulcs*nak nevezzük. A nem szűkíthető szuperkulcs neve *kulcs*. (1. miatt van ilyen.)

Anomáliák

1. **Módosítási anomália**
   egy attribútum értéke több helyen szerepel, így több helyen kell módosítani (inkonzisztencia).
2. **Beírási anomália**
   hiányos adatok miatt valamit nem lehet bevinni (információvesztés).
3. **Törlési anomália**
   egy sor törlésével olyan információ is elvész, amire még szükség lenne (információvesztés).

Példa anomáliákra

Egy reláció oszlopai:

Tantárgyszám, tantárgynév, követelmény, oktatónév, oktatócím

- **Változtassuk meg az oktató címét.**
- **Mi történik, ha egy új tárgyhoz nincs oktató kijelölve?**
- **Mi történik, ha egy oktató kilép?**

Az anomáliák kiküszöbölése

A felsorolt anomáliák kiküszöbölhetők, a relációk olyan felbontásával (dekompozíció), melyek eredményei normalizált relációk.

1. Normálforma

A reláció 1NF értelemben normalizált, ha minden sorának minden mező értéke egy elemi érték.

(Újabb rendszerek lehetővé teszik a figyelmen kívül hagyását.)

2. Normálforma

A reláció 2NF értelemben normalizált, ha 1NF értelemben normalizált, és ha egy mezőérték meghatározásához összetett kulcs szükséges, egy másik mezőérték meghatározásához nem elegendő ennek egy része.

Példa 2. normálformára

cikkszám, ár, szállítási sorszám, cikkleírás

Anomália: ha az utolsó sort töröljük, elvész a cikkszám, cikkleírás közötti összefüggés.

Ok: az ár azonosításához szükséges a cikkszám, szállítási sorszám összetett kulcs, a cikkle1rás meghatározásához elég a cikkszám.

Megoldás: dekompozíció.

cikkszám, cikkleírás

cikkszám, ár, szállítási sorszám

3. Normálforma

A reláció 3NF értelemben normalizált, ha 2NF értelemben normalizált, és ha egy mezőérték sem függ olyan attribútum értéktől, vagy ezek olyan halmazától, mely nem kulcs.

Példa 3. normálformára

rendszám, gyártmány, típus, gyártási idő, szín

Ha egy időben csak egy színt használnak, elvész a gyártási idő és a szín közötti összefüggés.

Boyce-Codd (BCNF) normálforma

A 3NF tovább gondolása.

A reláció BCNF értelemben normalizált, ha 3NF értelemben normalizált, és ha egy összetett kulcs egy részét sem határozza meg más attribútum érték (nincs kulcstörés).

Ha egy reláció BNCF, akkor 3NF is.

Példa BCNF normálformára 1.

tantárgy, tanár, diák

Tegyük fel, hogy egy tanár csak egy tárgyat oktat.
Mi legyen a kulcs?

Egyszerű kulcs nem egyértelmű.

tantárgy, tanár nem határozza meg a diákot

tanár, diák nem 2NF (a tanár meghatározza a tantárgyat).

tantárgy, diák nem BCNF (a tanár meghatározza a tantárgyat).

Példa BCNF normálformára 2.

Anomália: törléskor eltűnik, hogy egy tanár milyen tárgyat tanít.

Megoldás: dekompozíció

tantárgy, <u>tanár</u>
tanár, <u>diák</u>

Többértékű függőség

Az $A_1A_2...A_n \rightarrow\rightarrow B_1B_2B_m$ többértékű függőség fennáll, ha

az R reláció soraiból tekintve azokat, melyek minden A_i értéken megegyeznek, ezek Bj-ken felvett értékei függetlenek az A-któl és B-ktől különböző attribútumok értékeitől.

A többértékű függőség nem triviális, ha

1. **egyik B sincs az A-k között,**
2. **az A-k és a B-k együttesen sem tartalmazzák R összes atribútumát.**

4. Normálforma

Egy reláció 4NF-ben van, ha valahányszor az $A_1A_2...A_n \rightarrow\rightarrow B_1B_2B_m$ nem triviális többértékű függőség fennáll, akkor $\{A_1A_2...A_n\}$ szuperkulcs.

Ha a reláció 4NF, akkor BNCF is.

Példa többértékű függőségre I.

Pl. legyenek az R reláció oszlopai:

Tanár Tantárgy Kar

Egy tanár több tárgyat taníthat, egy tárgyat többen taníthatnak, egy tárgyat több karon tanítanak, egy karon több tárgyat tanítanak.

Tanár$\rightarrow\rightarrow$Tantárgy és Tantárgy$\rightarrow\rightarrow$Kar fennáll, és nem triviális.

Példa többértékű függőségre II.

Tanár	Tantárgy	Kar
...		
Nagy	AB	KGK
Nagy	AB	NIK
Kiss	PPT	NIK
...		

BCF, de nem redundancia mentes:

pl. egy Tanár → Tantárgy bejegyzés többször is előfordulhat.

Tanár Tantárgy nem szuperkulcs, azaz nem 4F.

Példa többértékű függőségre III.

Megoldás itt is a dekompozíció:

Tanár Tanfolyam

Tanár Kar

Tárgy Kar

Vigyázat:

Tanár Tárgy

Tanár Kar

nem jó, mert elvész, hogy melyik tanár melyik karon mit tanít.

További normálformák

Létezik 5NF, de kisebb jelentősége miatt nem foglalkozunk ezzel.

A lekérdezés elve relációs rendszerekben

1. relációs algebra
2. relációs kalkulus

A relációs algebra 1.

Alapműveletek:

1. **Unió:** $R \cup S$
2. **Különbség:** $R - S$
3. **Direkt szorzat:** $R \otimes S$
4. **Projekció:** $\Pi_{i1,i2...ik}(R)$
5. **Szelekció:** $\sigma_F(R)$

ahol: Az F feltétel az [i] Θ [j], és az [i] Θ c elemi kifejezések logikai műveletekkel ($\wedge$, $\vee$, $\neg$) való összekötéséből állhat. Θ tetszőleges összehasonlító operátort (=, ≠ , < , > , ≤ , ≥) jelenthet.

A relációs algebra 2.

Következmény műveletek:

6. Metszet: $R \cap S$

$R \cap S = R - (R - S)$

7. Hányados: $R \div S$

Feltesszük, hogy $r > s$, és $s \neq 0$.)

Legyen

$T = \Pi_{1,2,\ldots r-s}(R)$, **továbbá**

$V = \Pi_{1,2,\ldots r-s}((T \otimes S)-R)$.

Ekkor belátható, hogy

$R \div S = T - V$.

A relációs algebra 3.

8. Feltételes kapcsolat (Θ join):

$$R \underset{[i]\,\Theta\,[j]}{*} S = \sigma_{[i]\,\Theta[r+j]}(R \otimes S)$$

9. Természetes kapcsolat (natural join): R * S

Hasonló a feltételes kapcsolathoz de itt a szelekció feltétele az, hogy az azonos nevű oszlopokban azonos érték szerepeljen.

Példa relációs algebrai műveletekre 1.

Számoljuk ki az R ÷ S műveletet eredményét!

$R =$

a	b	c	d
a	b	e	f
b	c	e	f
e	d	c	d
e	d	e	f
a	b	d	e

$S =$

c	d
e	f

Példa relációs algebrai műveletekre 2.

Számoljuk ki az $R *_{B<D} S$ művelet eredményét!

R =

A	B	C
1	2	3
4	5	6
7	8	9

S =

D	E
3	1
6	2

Példa relációs algebrai műveletekre 3.

Számoljuk ki az R * S művelet eredményét!

R =

A	B	C
a	b	c
d	b	c
b	b	f
c	a	d

S =

B	C	D
b	c	d
b	c	e
d	d	b

Példa relációs algebra alkalmazására 1.

Tekintsük az alábbi három relációt.

a.) Autók: jele A

Mezői: rendszám, gyártmány, típus, szín, ...

b.) Emberek: jele E

Mezői: szemszám, név, foglalkozás, cím, ...

c.) Kapcsolat: jele K

Mezői: forgrsz, szemszám.

Feladat: **keressük ki a sárga opelek tulajdonosainak foglalkozását.**

Példa relációs algebra alkalmazására 2.

A megoldás:

$$R1 = \sigma_{\text{szín} = \text{sárga} \wedge \text{típus} = \text{opel}} (A)$$

$$R2 = \Pi_{\text{rendszám}} (R1)$$

$$R3 = R2 * K_{\text{rendszám}=\text{forgrsz}}$$

$$R4 = \Pi_{\text{szemszám}} (R3)$$

$$R5 = R4 * E$$

$$R6 = \Pi_{\text{foglalkozás}} (R5)$$

Példa relációs algebra alkalmazására 3.

A megoldás egy lépésben:

$$\Pi_f(\Pi_{szsz}(K *_{fsz=rsz} (\Pi_{rsz} (\sigma_{sz = sárga \wedge t = opel} (A)))) * E)$$

(foglalkozás→f, szemszám →szsz, forgrsz →fsz, rendszám →rsz, szín →sz, típus →t helyettesítéssel).