Adatbázis-kezelő rendszerek I.

INDEXEK ÉS TRANZAKCIÓKEZELÉS

Fogarassyné Vathy Ágnes Pannon Egyetem

Indexek

Indexek létjogosultsága

Az adatelérés a memóriából gyors, DE nagy adatbázisok esetén az összes rekord nem tartható egyidőben a memóriában.

- Index nélküli tárolás: kupac szervezés
  - **Keresés**: lineáris, nem hatékony
  - **Beszúrás**: a fájl végére
  - **Módosítás**: 1 keresés + módosítás
  - **Törlés**:
    - 1 keresés + a törölt rekord „deleted”-re állítása, vagy
    - 1 keresés + 1 törlés (üres hely marad)
  - Gyakori adatmanipulációknál időszakos újraszervezést igényel.
  - Hatékony tömeges adatfeltöltésnél.
- Index:
  - A táblákban való keresés és a sorbarendezés gyorsítására alkalmas eszköz.
  - Tervezés során nagy hangsúlyt kell rá fektetni.

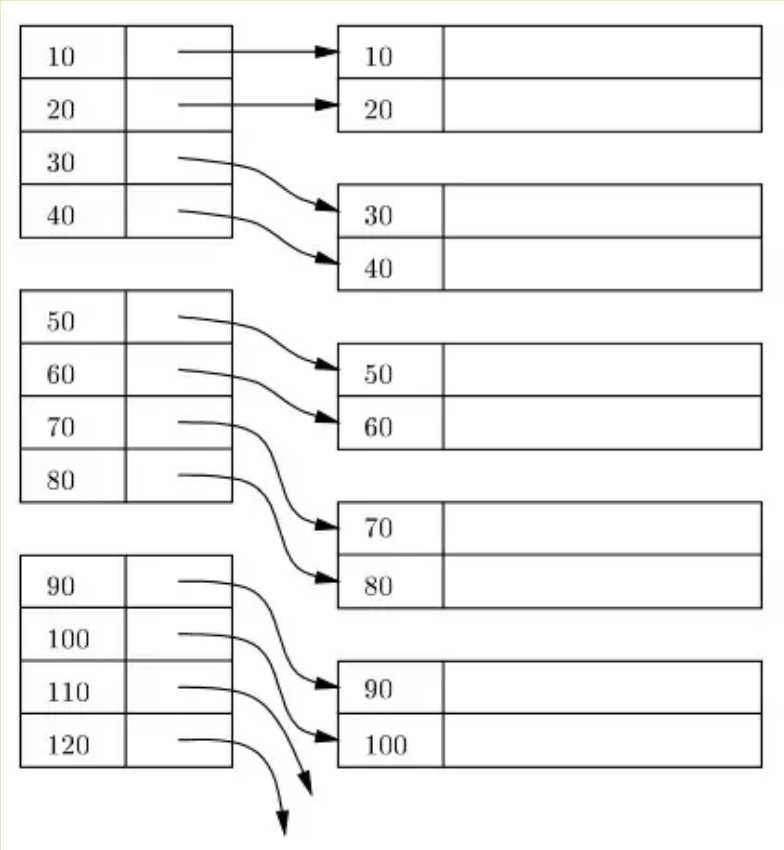
Index – index file

- Index bejegyzés: <index kulcs, rekord pointer>
- Index file: index bejegyzések konkatenációja (az elején egy extra pointerrel)
  - Az index file-ok jellemzően sokkal kisebbek, mint az adatfájlok
- Elsődleges index (clustered index): az adatfájl rendezettségének alapját adja
  - az adatblokkok az indexértékek alapján rendezettek
  - jellemzően: PRIMARY KEY
  - egyéb esetben UNIQUE korlátozás
- Másodlagos indexek (non-clustered index)
  - független a fizikai tárolás sorrendjétől
  - lehet több is

Elsődleges index vs. Másodlagos index

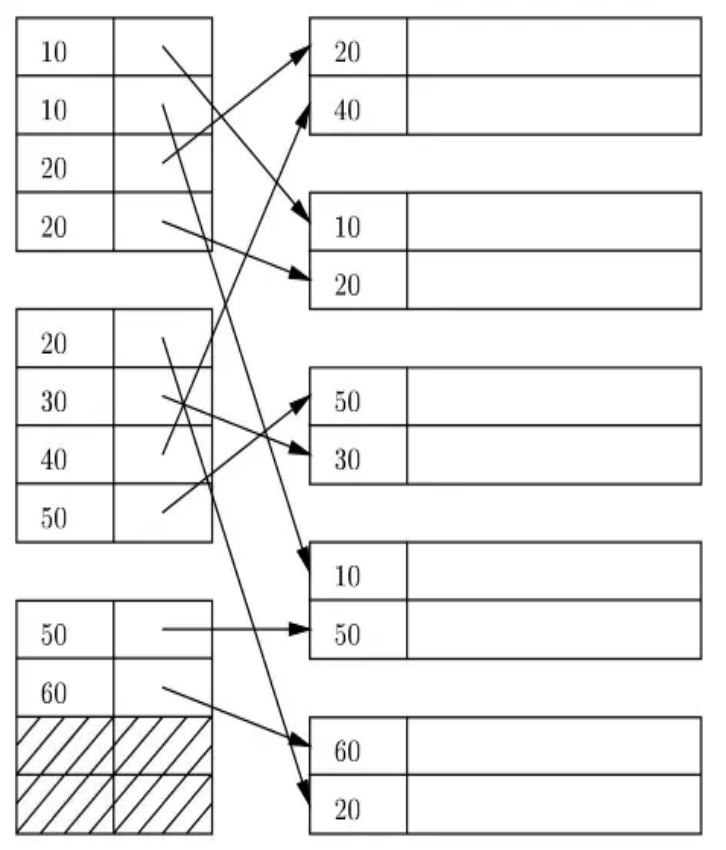
- Elsődleges index

index blokk adat blokk



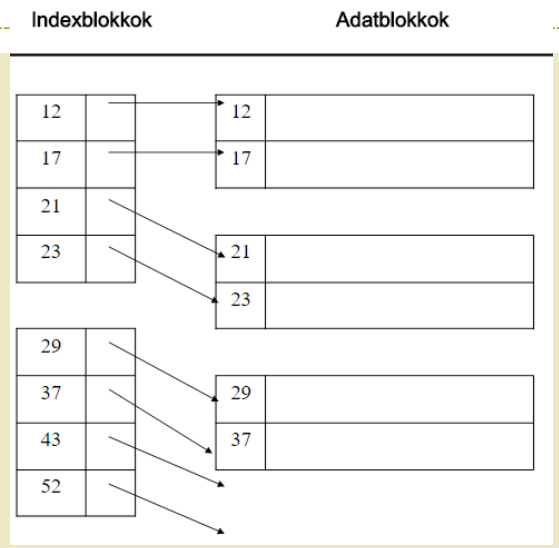
- Másodlagos index

index blokk adat blokk

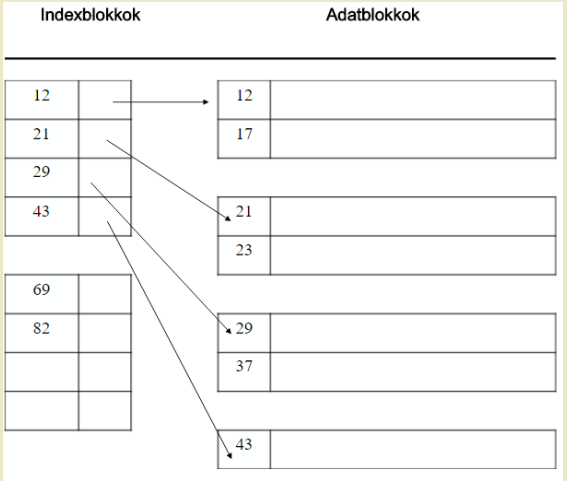


Index: ritka és sűrű indexek

- Sűrű indexek:
  - az adatállománynak nem kell rendezetten tárolni az adatokat
  - az adatfájl minden egyes rekordjához egy index-rekord tartozik



- Ritka index:
  - csak rendezett adatok esetén használható
  - csak az adatfájl blokkjának elején lévő kulcs-értékeket tünteti fel

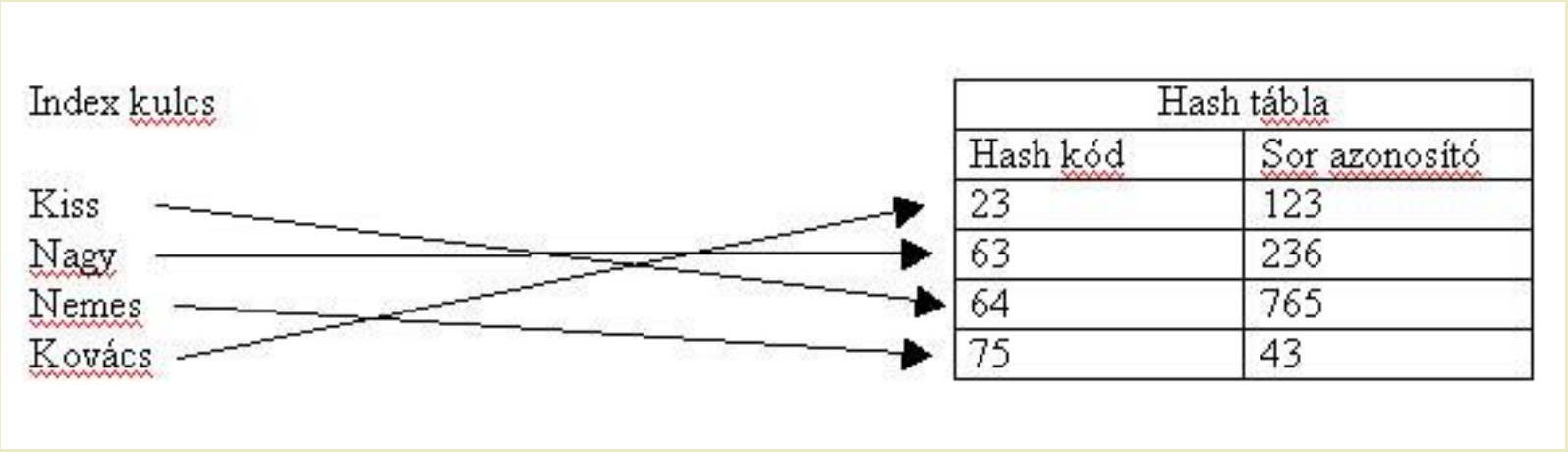


Indexek megvalósítása

- Indexek kialakítása leggyakrabban:
  - Hash tábla
    - Napjainkban már kevésbé használatos. Inkább csak hálós és hierarchikus rendszerek alkalmazták.
  - Bináris fa, B-fa, B+ fa, B* fa, R-fa, …

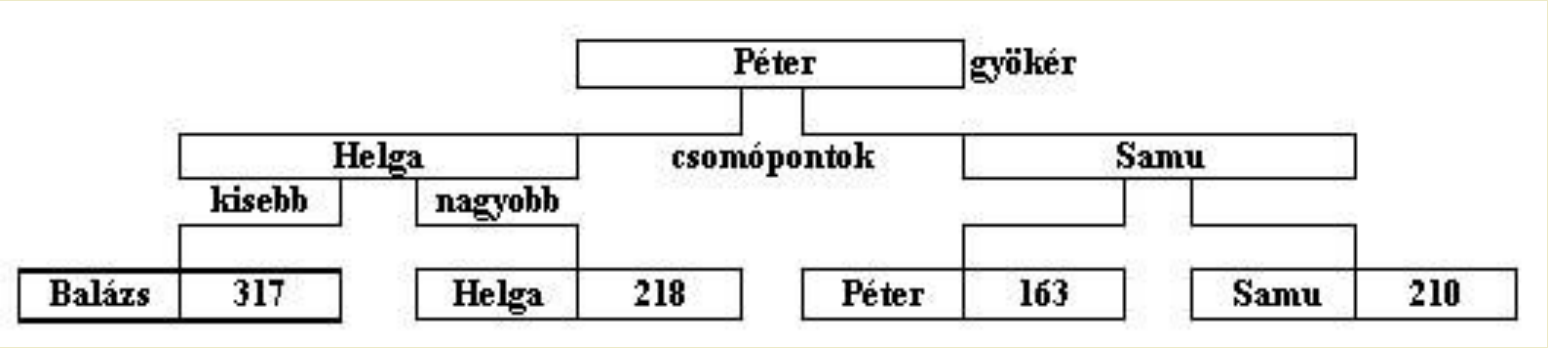
Index – Hash tábla

- Hash kód számítása matematikai alapokon az index mező értékéből.
  - pl.: prímszámmal történő osztás maradéka
- Hash tábla: tartalmazza a keresett érték fizikai címét.
- Hash kód ütközés esetén: azonos kódot adó kulcsok láncolása listába.



Index – Bináris fa

- Index kulcsok növekvő vagy csökkenő sorrendbe rendezése, majd bináris fa építése.
- A fa gyökere és csomópontjai nem tartalmazzák az index kulcshoz tartozó sor fizikai helyét, hanem csak a fa levelei.
- A keresés mindig a gyökértől kezdődik, a megfelelő ág felé folytatódik, és akkor ér véget, ha egy levélhez érünk.

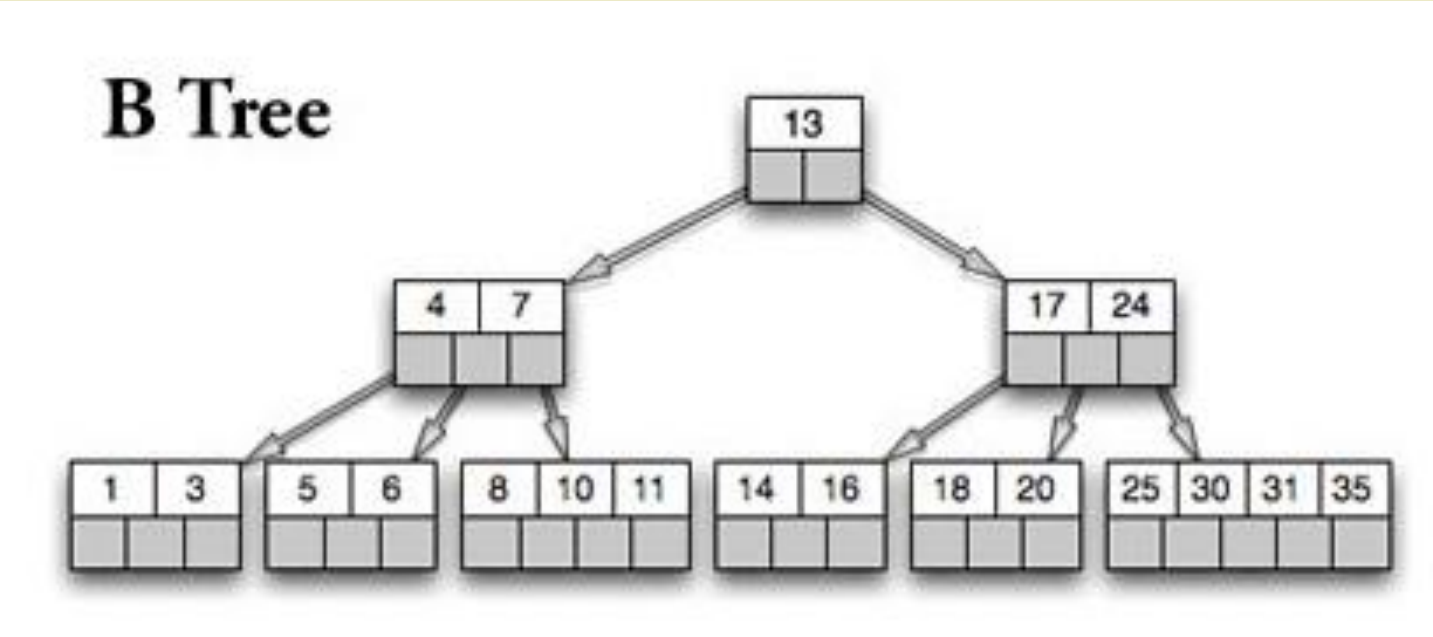


- Nem túl hatékony, mert adatmanipulációs műveletek esetén gyakran újra kell építeni a fát.

Index – B-fa

A keresőfák felépítésénél arra törekszenek, hogy a fa valamennyi ága azonos hosszúságú legyen.

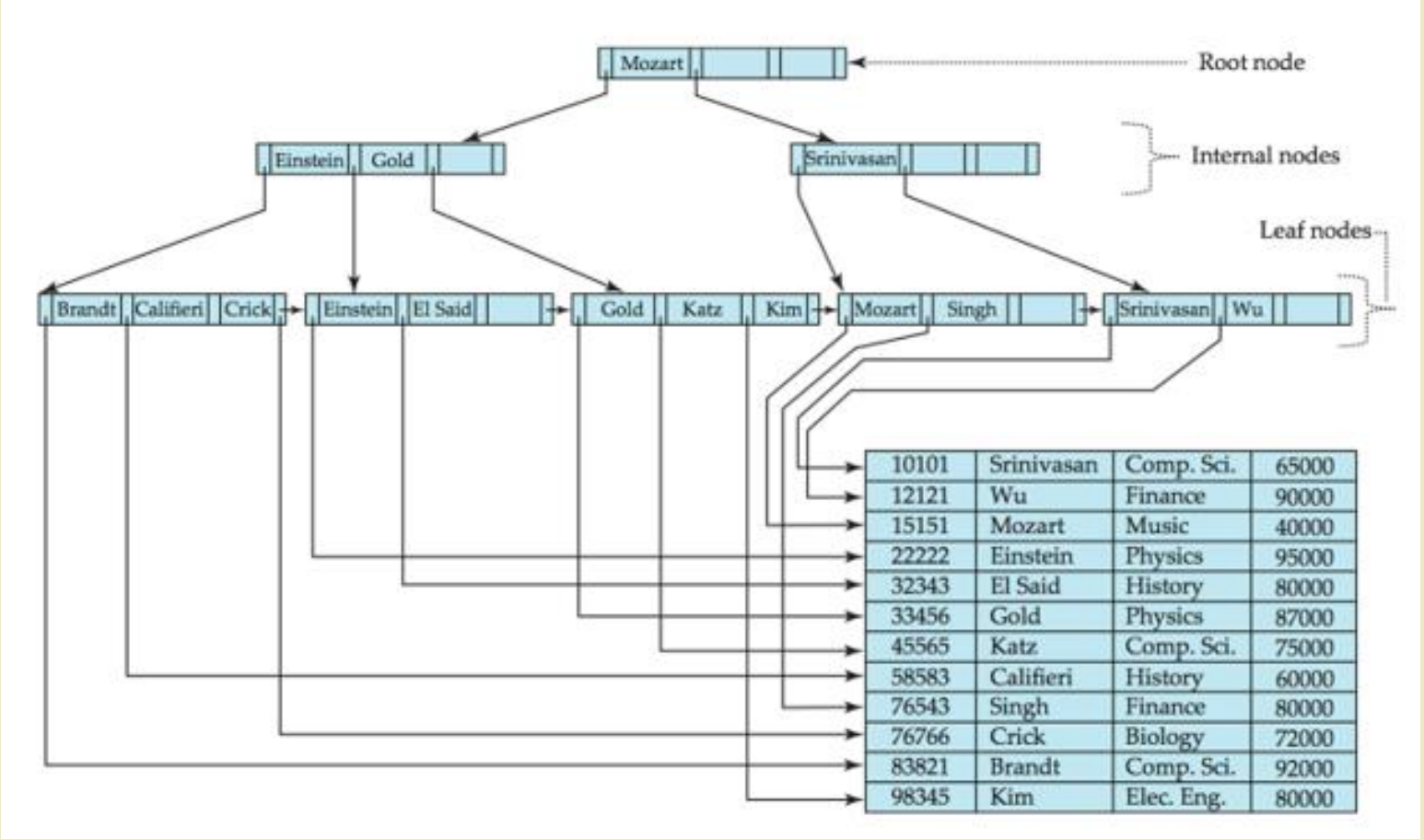
→ B-fa (*Balanced?* Tree)



B-fa vs. B+ fa

- B-fa:
  - A köztes csúcsok is tartalmazhatnak <index kulcs, rekord pointer> párokat.
  - A B-fa levelei egymással nincsenek összekapcsolva.
    - A kulcsokon történő lineáris keresés során a fa minden ágán végig kell utazni
- B+ fa:
  - Nem tárolnak adat pointereket a belső csúcsokon, a mutatók csak a levélszinten jelenhet meg.
    - =>a belső csúcsok több kulcsértéket tudnak tárolni a memóriában
  - A B+ fa levelei egymással kapcsolva vannak.
    - A kulcsokon történő lineáris keresés során csak 1-szer kell végigmenni a leveleken.
  - Leggyakrabban alkalmazott megvalósítás.

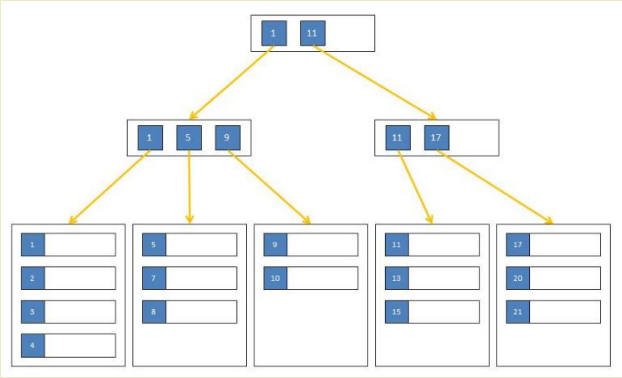
B+ fa


Mozart
Root node
Einstein
Gold
Srinivasan
Internal nodes
Leaf nodes
Brandt
Califieri
Crick
Einstein
El Said
Gold
Katz
Kim
Mozart
Singh
Srinivasan
Wu
10101 Srinivasan Comp. Sci. 65000
12121 Wu Finance 90000
15151 Mozart Music 40000
22222 Einstein Physics 95000
32343 El Said History 80000
33456 Gold Physics 87000
45565 Katz Comp. Sci. 75000
58583 Califieri History 60000
76543 Singh Finance 80000
76766 Crick Biology 72000
83821 Brandt Comp. Sci. 92000
98345 Kim Elec. Eng. 80000

Indexek kialaktása

- 1-1 index kialakítása:
  - Történhet 1 vagy, több mező alapján (1 mezős és többmezős indexek)
  - Gyorsítja: Az indexelt adatok keresését.
  - Lassítja: általánosan a beszúrás, törlés és bizonyos esetekben a módosítás műveletét

  - Másodlagos indexek esetében levélszinten: mutatók az adatrekordokra
  - Elsődleges indexek esetében levélszinten: adatrekordok



Tranzakció kezelés

Tranzakció kezelés

- A DBMS támogatja a több felhasználós környezetet, és az adat megosztását:
  - egymással versengő felhasználók
  - inkonzisztens adatok?
- Tranzakció: az adatbázisműveletek 1 végrehajtási egysége
  - A DBMS biztosítja: minden egyes **tranzakció** vagy tökéletesen végrehajtódik (COMMIT), vagy az eredeti állapot állítódik vissza (ROLLBACK).
    - **Tranzakció kezelő (TM)**: Az adatbázis alkalmazások fontos része, mely lehetővé teszi egymással konkuráló tranzakciók százainak egyidejű végrehajtását.
    - A **tárkezelő** biztosítja, hogy minden egyes teljesített tranzakció esetén a végrehajtott módosítások az adatbázisban tartósan rögzüljenek.

Tranzakciók ACID tulajdonsága

- ACID tulajdonságok:
  - Atomosság (Atomicity): a tranzakció „mindent vagy semmit” jellegű végrehajtása (vagy teljesen végrehajtjuk, vagy egyáltalán nem hajtjuk végre).
  - Konzisztenciamegőrzés (Consistency preservation): a tranzakció végrehajtása után is teljesüljenek az adatbázisban előírt konzisztenciamegszorítások (integritási megszorítások).
  - Elkülönítés (Isolation): minden tranzakciónak látszólag úgy kell lefutnia, mintha ez alatt az idő alatt semmilyen másik tranzakciót sem hajtanánk végre.
  - Tartósság (Durability): az a feltétel, hogy ha egyszer egy tranzakció befejeződött, akkor már soha többé nem veszhet el a tranzakciónak az adatbázison kifejtett hatása.

Párhuzamosan futó tranzakciók hatásai, problémái

- Piszkos olvasás: Amennyiben párhuzamosan futó tranzakciók láthatják egymás nem commitált adatait, akkor olyan adatokkal dolgozhatnak, melyeket később a másik tranzakció visszagörget.
- Elveszett módosítás: Ha két vagy több tranzakció ugyanazon az adatelemen dolgozik, akkor a tranzakciók felül tudják írni egymás módosításait es csak annak a tranzakciónak marad meg a hatása, mely utolsónak adta ki a commit utasítást.
- Nem megismételhető olvasás: Egy tranzakció élete során egy rekordot nem lehet állandónak tekinteni, mert azt egy másik tranzakció módosíthatja. Így ha egy rekord lekérdezését megismételjük a tranzakcióban többször is, nem biztos, hogy mindig ugyanazt az eredményt kapjuk.
- Fantom rekordok: A fantom rekordok olyan rekordok, melyek megjelenhetnek illetve eltűnhetnek egy táblából egy tranzakció élete során egy másik párhuzamosan futó tranzakció működése révén.

ACID elvek biztosítása

- ACD:
  - tranzakciós logfálok
  - adatbázis mentések
- I:
  - Különféle szintű zárolások

Zárolások

Zárolás: a tranzakció lefoglalja az objektumot, amit használni szeretne

kevesebb konfliktus
nagyobb adminisztráció

több konfliktus
kisebb adminisztráció

- Zárolási paraméterek:
  - **Objektum mérete**:
    - rekord, lap, kulcs, index, tábla, adatbázis kis méret … nagy méret
  - **Zárolás időtartama**:
    - Objektum lefoglalása
    - Objektum feloldása
  - **Művelet**:
    - Írás: **exclusive lock** (kizárólagos hozzáférés, más nem férhet hozzá)
    - Olvasás: **shared lock** (osztott hozzáférés, csak olvashatja)
    - Módosítás: **updale lock** (előjegyzett kizárólagos)

TM-enként eltérő implementációk!

Kétfázisú zárolás

Kétfázisú zárolási protokoll (2PL):

- Kiterjesztési fázis: szükséges zárak elhelyezése (de egyetlen sem kerül elengedésre).
- Feloldási fázis:
  - A zárolások feloldhatók, de újabb zár már nem helyezhető el
  - a tranzakció befejezésekor az összes zár feloldásra kerül.

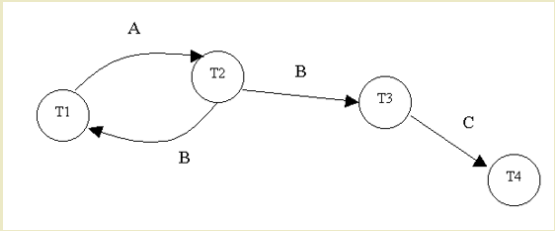
- Különféle típusai léteznek

Legtöbb rendszer 2PL-lel biztosítja az ACID elvek betartását.

Holtpont

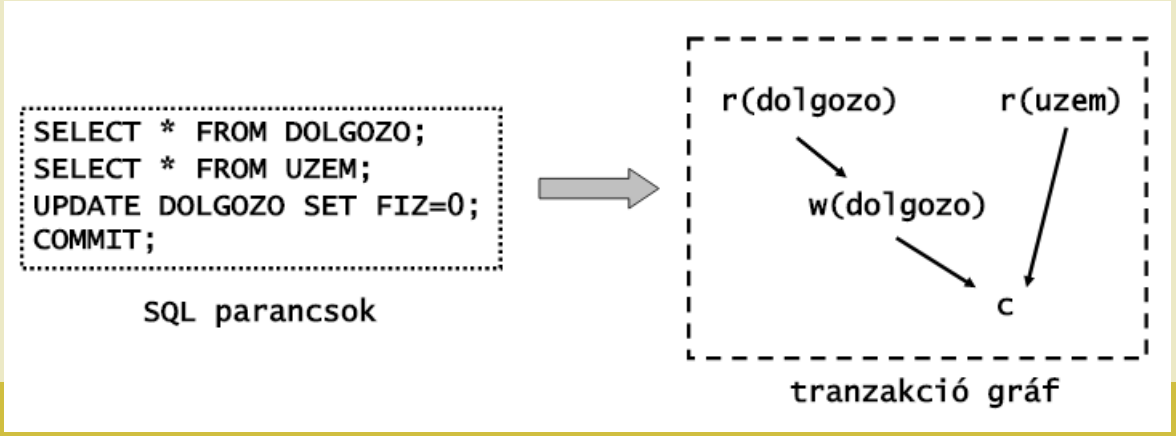
Holtpont: olyan állapot, mikor két vagy több tranzakció vár a másik által lezárt objektumra.

- Holtpont feloldása:
  - Módosítási zárak alkalmazása (update lock)
  - Várakozási gráf
    - csúcsok: tranzakciók
    - élek: T1 -> T2, ha:
      - T1 lezárva tartja az ”x” adatbáziselemet.
      - T2 kéri az ”x” adatbáziselemet, hogy zárolhassa azt.
    - holtpont = hurok a gráfba
    - holtpont feloldása: kiválasztott tranzakció (áldozat, victim) visszagörgetése
  - Időkorlát mechanizmus (timeout)



Tranzakció gráf

- Tranzakció gráf:
  - irányított gráf, ahol
    - csúcsok: műveletek (read (r), write (w))
    - élek: megelőzési relációk
- Megkötések:
  - Gráfnak tartalmazni kell egy lezárási műveletet (commit (c), abort (a))
    - minden tranzakcióra pontosan 1 művelet,
    - csak a legutolsó elemként jelenhet meg

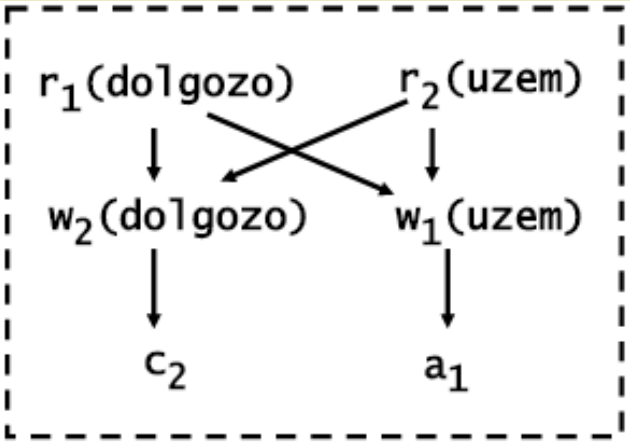


History

- History: a rendszerben futó tranzakciók összessége
- History gráf:
  - + szerepel a tranzakció azonosítója is

```
T1: SELECT * FROM DOLGOZO;
T2: SELECT * FROM UZEM;
T2: UPDATE DOLGOZO SET FIZ=0;
T1: DELETE FROM UZEM;
T2: COMMIT;
T1: ROLLBACK;
```

SQL parancsok



history gráf