Alapvető fizikai tárolási szerkezetek, indexek, B-fák

Takács Gábor
mérnök informatikus, okl. mérnöktanár
takacsg@sze.hu
http://rs1.sze.hu/~takacsg/

Adatelemek ábrázolása

Adatokat bájtok sorozatával tárolunk!

- **Számok:**
  - PONTOS számok: 2 vagy 4 bájt (smallint, int)
  - KÖZELÍTŐ számok 4 vagy 8 bájt (float, real)
- **Rögzített bosszú karakterláncok**
  helykitöltő töltelékkarakter használata (pad char: ” ¬ ”)
  CHAR(n)
    - *n* hosszúságú karakterláncot reprezentál
    - *n* bájtból álló tömbben tárolódik

  Pl.: CHAR(6) értéke ”kapu”, akkor így tárolódik: ’kapu¬¬
- **Változó bosszú karakterláncok**
  VARCHAR(n)
- **UNICODE karakterek:** nchar, nvarchar, ntext

A ténylegesen szükséges tároló hely a konkrét adat függvényében változik (n+1 bájtot rendelünk a karakterlánchoz)

Adatelemek ábrázolása

Adatokat bájtok sorozatával tárolunk!

- **Dátumok és időpontok**
  DATE: 10 hosszú karakterlánccal ábrázoljuk
  - EEEE-HH-NN Pl.:'2013-11-25'

  TIME: 8 hosszú karakterlánccal ábrázoljuk
  - OO:PP:MM.másodperctörtrésze (számjegy)

  DATETIME: kettő együtt történő tárolása

- **Bitek**

  Bitsorok **BTI(n) –** 8 bitet 1 bájtba foglalunk
  - Például, a 010111110011 ábrázolható úgy, hogy 01011111 az első bájt, és 00110000 a második bájt.
  - Logikai értékek: **Boolean érték**
    igaz: 1000000 vagy 11111111 (könnyebb ellenőrzés miatt)
    hamis: 00000000

Az adatok fizikai szervezése

Mi a célja a fizikai szervezésnek?

- A **lekérdezés**ek **hatékony**abb (gyorsabb) végrehajtása.
- A végrehajtási idő a lekérdezés eredményének méretével legyen arányos és ne a kiindulási reláció méretével.

Az ABKR működése közben hol helyezkednek el az adatok?

- Cache memória, RAM, Másodlagos tárolók – lemezek, Harmadlagos tárolóeszközök

Alapelv: A gyakran használt adatokat próbáljuk meg a gyorsabb elérésű helyen tartani.(pl. bizonyos táblázatokat a memóriában)

A fizikai adatmodell elemei:

- Adatállományok (fájlok)
- Blokkok
- Rekordok
- Mező

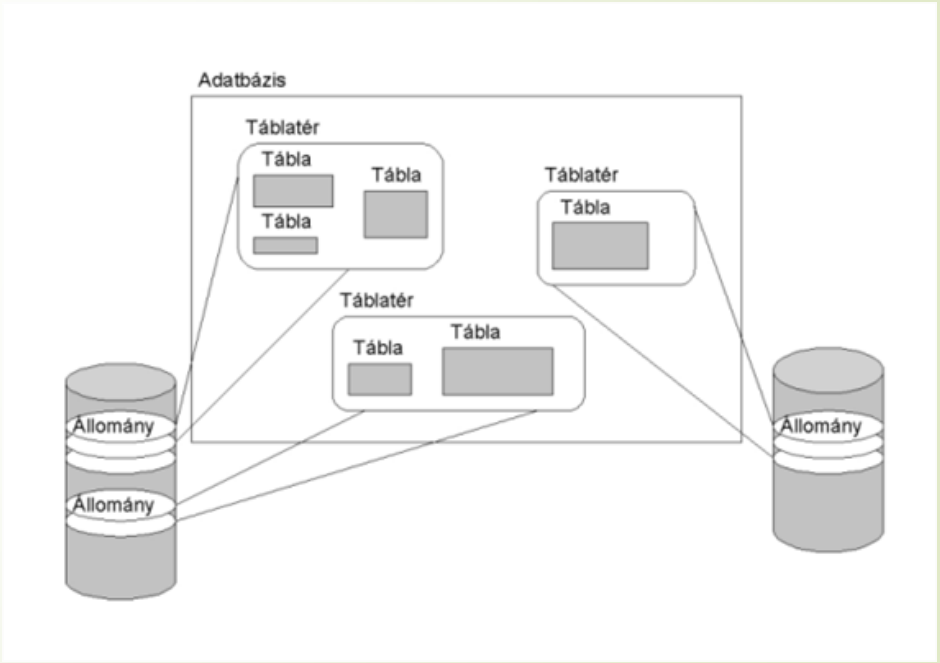
Táblaterek (logikai struktúra)

A relációs **adatokat** a felhasználók **logikailag táblákba szervezetten** tudják kezelni.

Azonban ezeknek az adatoknak állományokban kell megjelenniük a háttértárolón.

- **A fizikai tervezéskor** az adatbázis-adminisztrátor a táblákhoz egy olyan logikai struktúrát rendel, amely **azt fogja meghatározni, hogy a tábla adatai milyen fizikai állományokban lesznek tárolva**. Ezeket a logikai struktúrákat általában táblatereknek hívjuk
- Táblaterek szervezése függ a táblatér típusától, és az adatbázis-kezelő rendszer tulajdonságaitól
- A táblatérhez meg lehet adni a **tárolásra vonatkozó paramétereket is**, mint például azt, hogy a **táblatérnek mekkorák legyenek az adatlapjai**, vagy esetleg a mérete dinamikusan változhat-e.

Állományok, táblaterek, és a táblák kapcsolata



Adatlap - adatrekord

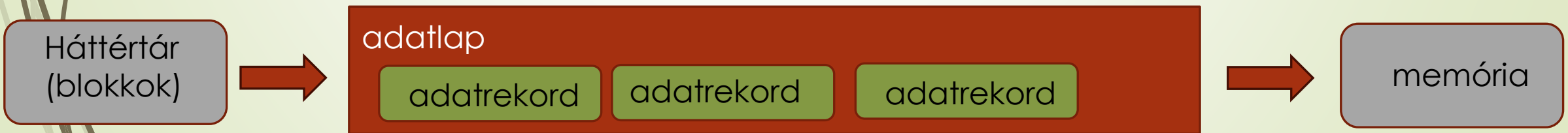
Az **adatlap az az adatmennyiség, amit az adatbázis-kezelő rendszer egyszerre tudd a memóriába beolvasni.**

- A lemez tárolási egysége az operációs rendszerbeli blokk.
- A blokkok mérete fix.

A lemezről az adatokat az adatbázis-kezelő rendszer a memóriába olvassa fel. Ennek a beolvasásnak az egysége az adatlap

adatlap mérete = a lemezblokk méretével
(vagy annak az egész számú többszörösével)

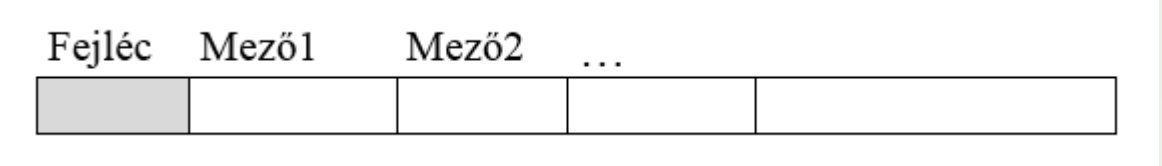
- A táblaterekhez rendelt állományok felépítésének alapja az adatlap.
- **A tábla egy-egy sorát az adatbázis-kezelő rendszer adatrekordba szervezi.**



Rekordok és mezők

- A relációk sémáit az adatbázis tárolja. A séma tartalmazza a rekord mezőinek neveit, adattípusait és a mezők kezdetét a rekordon belül.

Rekord felépítése:



A fejléc tartalma lehet:

- A rekord kezelésével kapcsolatos információk (pl.: törölt-e, melyik relációhoz tartozik stb.).
- A mezők típusa.
- Időbélyeg (mikor módosult utoljára).

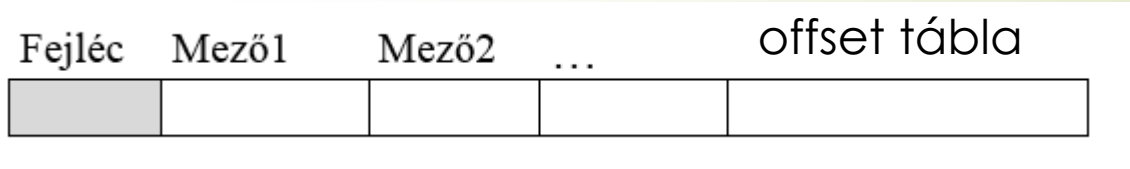
Egy mező felépítése:

Hossz	**NULL** jelző	**Érték**

Az **adatrekord** pontos felépítése:

- *fejléc*:
  Egy rekord általában néhány bájtnyi adminisztrációs információval kezdődik, amely a rekord adattartalmának szerkezetét és elrendezését határozza meg.

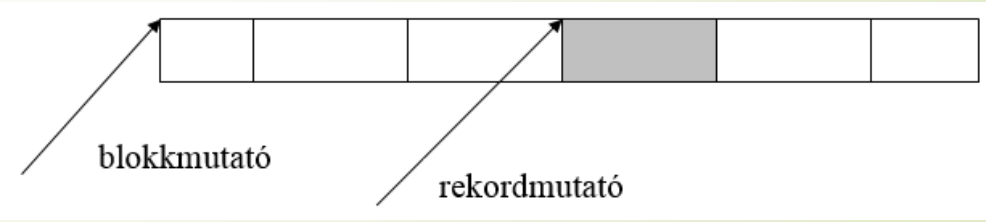
- *adatok*: A tábla egy sorának aktuális adattartalmát tartalmazzák az oszlopdefiníció sorrendjében.



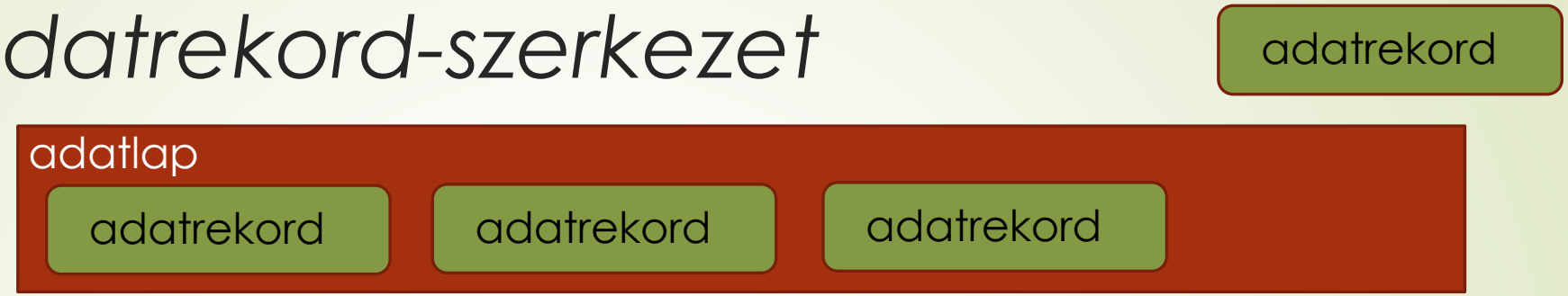
- *opcionális offset tábla*: a rekord tartalmazhat mutatókat, hogy a rekordban tárolt változó hosszúságú mezőket kezelje és felügyelje.

Mutatók:

Mutathat rekordra vagy blokkra.
(ami a blokk vagy rekord abszolút címét jelenti)



Adatrekord-szerkezet



Az adatrekordokat azonban egyaránt el kell helyezni a memóriában és az állományokban is.

- A memória és az állományok felépítésének az alapegysége az **adatlap**.
- **Azaz az adatrekord bele kell, hogy kerüljön az adatlapba.**

A legalapvetőbb kérdés az, hogy a kettő közül melyik a nagyobb?

Az esetek nagy részében **egy adatlapra több adatrekord kerülhet**, de nagyméretű adatrekordnál előfordulhat, hogy az nem fér rá egy adatlapra. Nagyméretű adatrekord akkor fordulhat elő, ha egy táblában olyan adattípusú adatokat tárolunk, mint szövegek, képek.

Ha más típusú rekordoknál is előfordul az, hogy az adatrekord nagyobb, mint az adatlap, akkor az adatbázis-adminisztrátor a telepítésnél, vagy a táblatér létrehozásánál nem jól választotta meg az adatlap méretét. Ez pedig teljesítményproblémákhoz vezethet.

Az adatok fizikai szervezése

Feltűzött (pinned) rekordok:

- Azok a rekordok, amelyekre valahonnan mutató mutat.
- Nem mozgathatók máshová, mert akkor rossz helyre fognak mutatni a mutatók.
- Nem törölhetők anélkül, hogy egy bejegyzést hagynánk a helyükön. Nem kerülhet másik rekord a helyükre

Rekordok típusa

Rögzített hosszú (között formátumú) rekordok
ha minden mező rögzített hosszú

Változó hosszúságú/formátumú rekordok:
ha van változó hosszú mezője.

Pl.: szöveget tartalmazó adattáblák
(vagy nagyméretű mezők képek, videók, stb.).

Rekordok típusa – **Rögzített hosszúságú rekord**

A rekordokat a lemez blokkjaiban tároljuk. A blokkokat az adatbázis-kezelő mozgatja. A blokk méretet oprendszer függő. Blokkméret n-szerese, ahol n=1,2,4,8Kb. A leggyakoribb blokkméret a 8Kbájt.

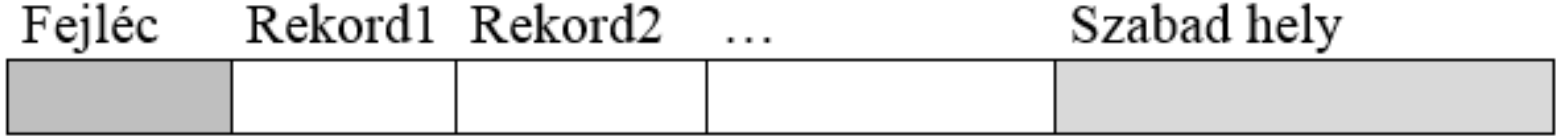
Példa: Tegyük fel, hogy a rekordjaink 416 bájt hosszúak, és 8K-s (8192 bájt) blokkokban tárolódnak.

Ebből 96 bájtot fogunk blokkfejlécre felhasználni. Hány rekordot helyezhetünk el a blokkban?

- A fejrész helyfoglalása után 8096 bájt marad az adatoknak.
- Az osztás (8096/416) elvégzése után adódik, hogy maximum 19 rekordot helyezhetünk el egy blokkban és marad szabadon 192 bájt.

Blokkok szervezése

Blokkfejléc + rekordok + szabad hely



Blokkfejlécben tárolt információk:

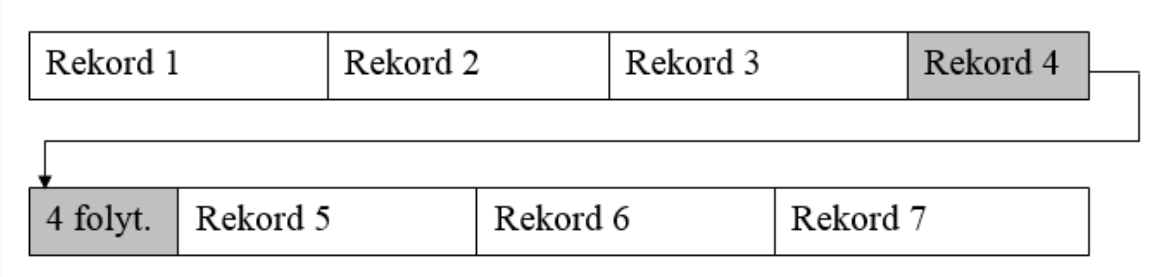
- Blokkbeli rekordok kezdőcímei
- Blokkbeli szabad hely kezdete
- Időbélyegzések (mikor módosították utoljára)
- Zárolási információk
- Lehet-e a blokkba még további sorokat tenniMutatók további blokkokra

Blokkokkal kapcsolatos további fogalmak

Blokkolási faktor: Bfr

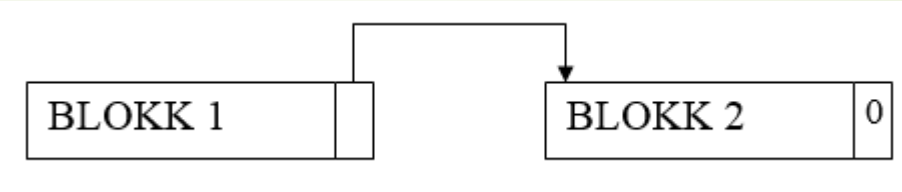
- Az egy blokkban elférő rekordok átlagos száma

Átnyúló rekord: Akkor lehet rá szükség pl. ha egy változó hosszúságú rekord a módosítás után már nem fér be a blokkba.



Túlcsordulási blokk:

Akkor lehet rá szükség, ha a 2. blokkbeli rekordoknak is az első blokkban kellene lenniük valamilyen szervezési módszer miatt.



Adatbázis-műveletek költsége

- Általában csak a blokk-olvasásokat és írásokat vesszük figyelembe. Feltételezzük, hogy minden blokk műveletnek azonos a költsége.

Költségbecslések jelölései:

n: A rekordok (sorok) száma

Bfr: Az egy blokkba átlagosan beférő sorok száma

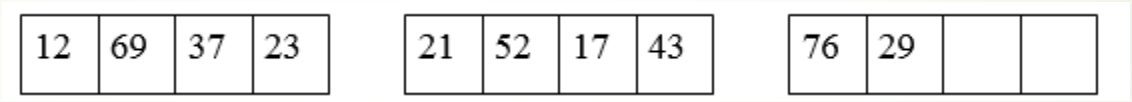
B: A tároláshoz szükséges blokkok száma (B = n/Bfr)

- Az alapvető műveletek, amelyeknek a költségét becsüljük:

Keresés Beszúrás Törlés Módosítás

Adatbázis-műveletek költsége
A Halom (heap) szervezés

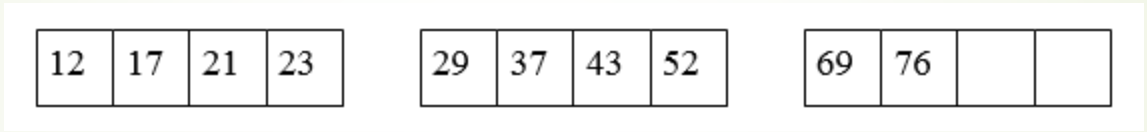
- A rekordok rendezetlenül (a beszúrások sorrendjében) vannak tárolva a blokkokban. A blokkok elhelyezése nem követ semmilyen speciális elvet.

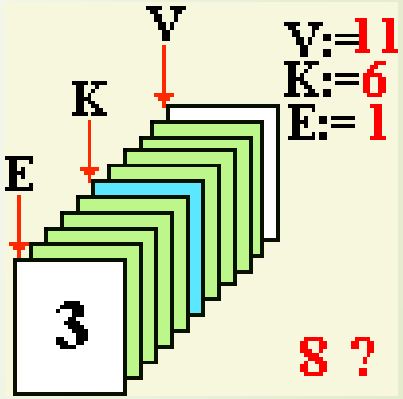


- **Keresés: n/2Bfr**
  (Lineáris kereséssel átlagosan a blokkok felét kell beolvasnunk)
- **Beszúrás: 2**
  (Elég a legutolsó blokkot beolvasni és kiírni)
- **Törlés: n/2Bfr +1**
  (Megkeressük a blokkot, majd a törlés után kiírjuk)
- **Módosítás: n/2Bfr + 1**
  (Megkeressük a blokkot, majd a módosítás után kiírjuk)

Adatbázis-műveletek költsége
Rendezett fájlok

- A kulcsértékük szerint rendezetten tároljuk a rekordokat.





- **Keresés:** $\log_2(B)$
  (Bináris kereséssel kereshetünk)
- **Módosítás:** $\log_2(B) + 1$
  (Megkeressük a blokkot, majd kiírjuk.
  Ha a kulcsmező módosul akkor törlés + beszúrás)
- **Törlés:** $\log_2(B) + 1$
- **Beszúrás:** A beszúrás nagyon költséges lehet, ha a rekord nem fér be a megfelelő blokkba mert a rendezettség fenntartásához sok rekordot arrébb kell mozgatnunk.
  Túlcsordulási blokkot alkalmazunk.

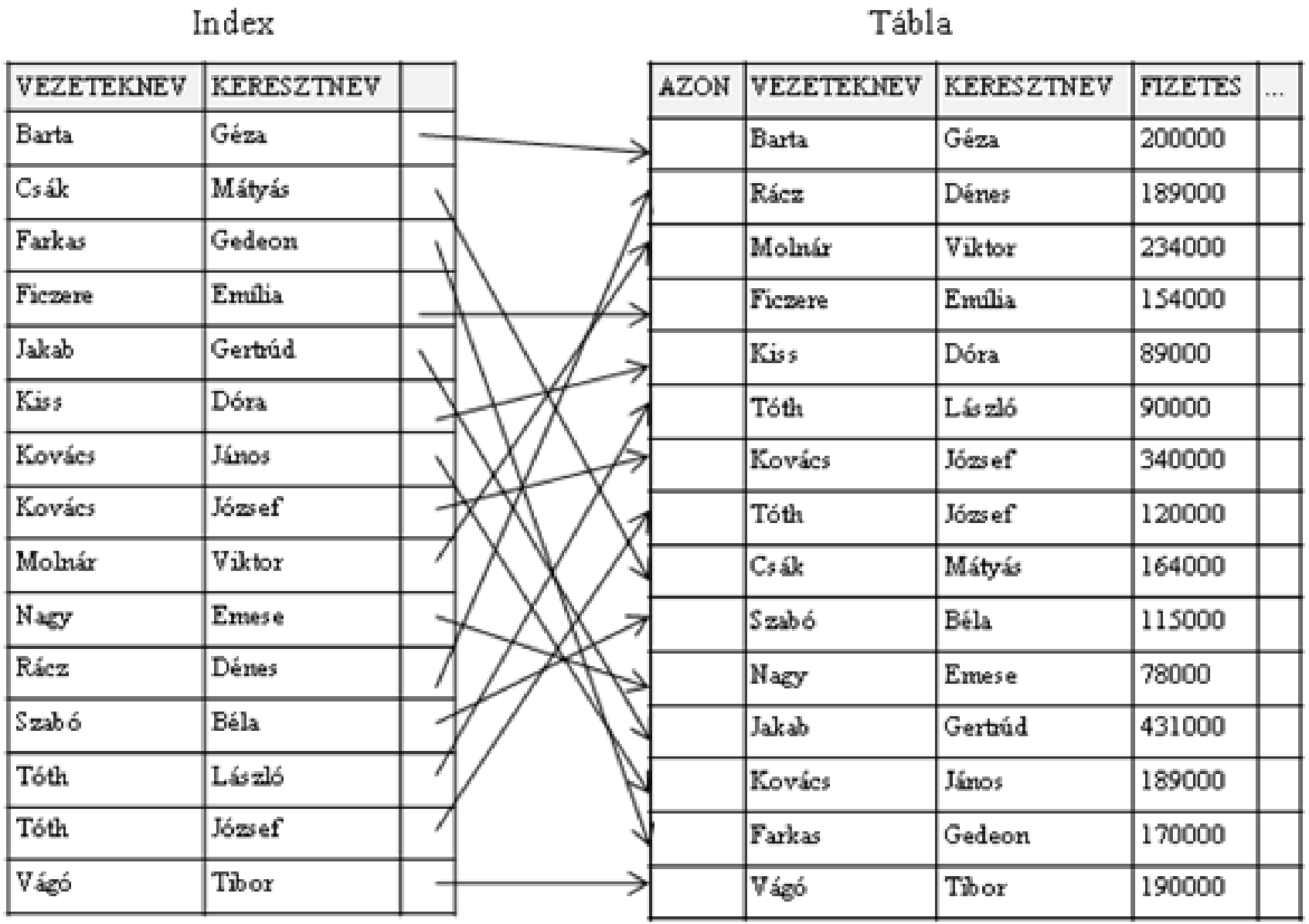
Index struktúrák

Tekintsünk egy egyszerű lekérdezést:

```
SELECT VEZETEKNEV, KERESZTNEV
FROM DOLGOZO
WHERE VEZETEKNEV=,Széchenyi';
```

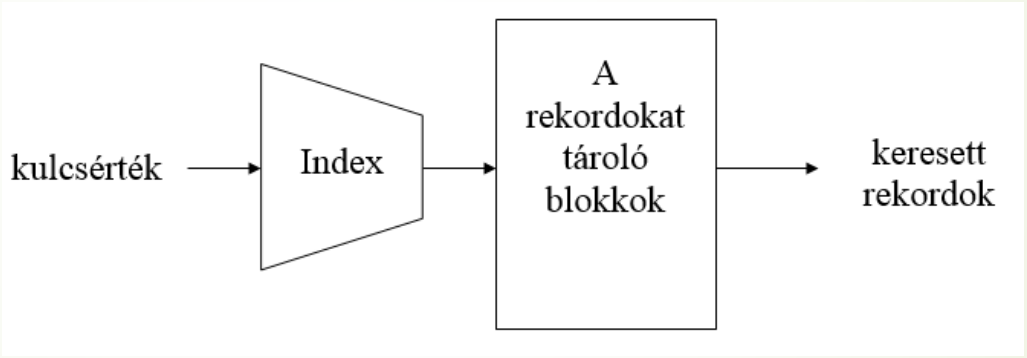
Az ilyen típusú keresések megkönnyítésére hozhatunk létre egy táblához egy vagy több indexet.

Az index segítségével az adatbázis-kezelő rendszernek nem kell a tábla minden sorát végignéznie, hanem annak csak egy töredék részét. Az index alapjául szolgáló oszlopokat indexkulcsoknak nevezzük.


Index
Tábla
VEZETEKNEV
KERESZTNEV
Barta
Géza
Csák
Mátyás
Farkas
Gedeon
Ficzere
Emília
Jakab
Gertrúd
Kiss
Dóra
Kovács
János
Kovács
József
Molnár
Viktor
Nagy
Emese
Rácz
Dénes
Szabó
Béla
Tóth
László
Tóth
József
Vágó
Tibor
AZON
VEZETEKNEV
KERESZTNEV
FIZETES
...
Barta
Géza
200000
Rácz
Dénes
189000
Molnár
Viktor
234000
Ficzere
Emília
154000
Kiss
Dóra
89000
Tóth
László
90000
Kovács
József
340000
Tóth
József
120000
Csák
Mátyás
164000
Szabó
Béla
115000
Nagy
Emese
78000
Jakab
Gertrúd
431000
Kovács
János
189000
Farkas
Gedeon
170000
Vágó
Tibor
190000

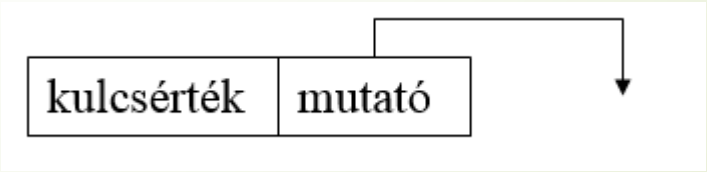
INDEX struktúrák

- Az előző két szervezési módnál csak magukat az adatrekordokat tároltuk. **Az index egy olyan további adatszerkezet, amelyik a rekordok gyorsabb megkeresését segíti elő.**



Az index mindig megadott kulcsértékű rekordok megkeresését segíti elő.

- Index állomány rekordjai:



- Kulcsérték több mezőből álló összetett érték is lehet.

INDEX struktúrák

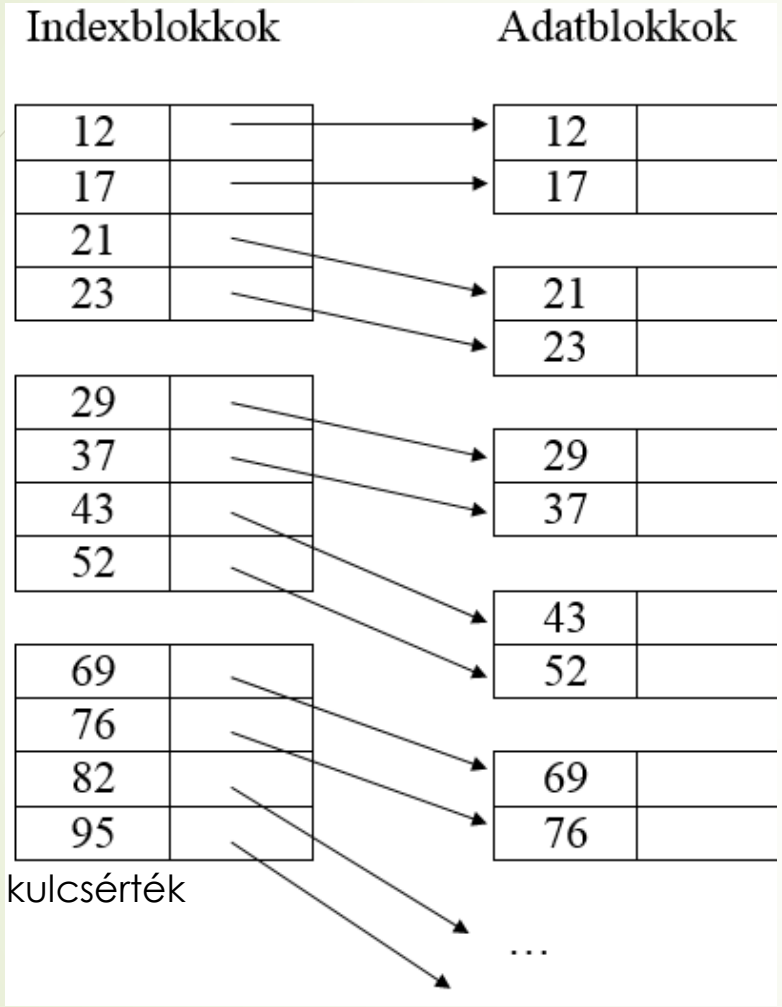
Leggyakoribb adatszerkezetek, melyeket indexként használhatunk

- Egyszerű indexek rendezett fájlokon
- Másodlagos indexek nem rendezett fájlokon
- Többszintű indexek, B-fák
- Hasítótáblázatok

Sűrű index és ritka index fogalmak:

- **Az INDEX SŰRŰ,** ha **minden adatrekordra** vonatkozóan tartalmaz egy (kulcs, mutató) típusú bejegyzést.
- **Az INDEX RITKA**, ha **csak bizonyos adatrekordokra** vonatkozóan tartalmaz (kulcs, mutató) típusú bejegyzést. **Ezek általában a blokkok első rekordjai.**

Példa: SŰRŰ indexre

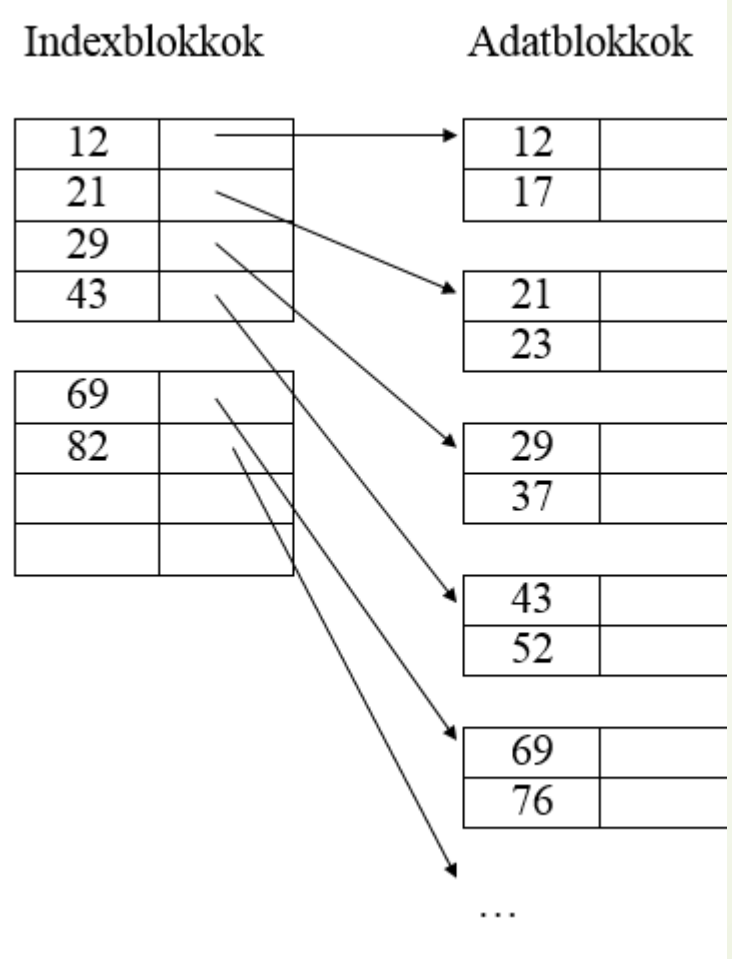


Miért jó, ha ugyanannyi állomány van?

- Index rekordok kisebbek
- Index rekordok mindig kulcs szerint rendezettek
- Ha az index elég kis méretű, akkor a leggyakrabban használt blokkokat, vagy akár az egész indexet a memóriában tarthatjuk.

Az indexnek ugyanannyi rekordja van, mint az adatállománynak.
Az adatállománynak nem kell rendezetten tárolnia a rekordokat

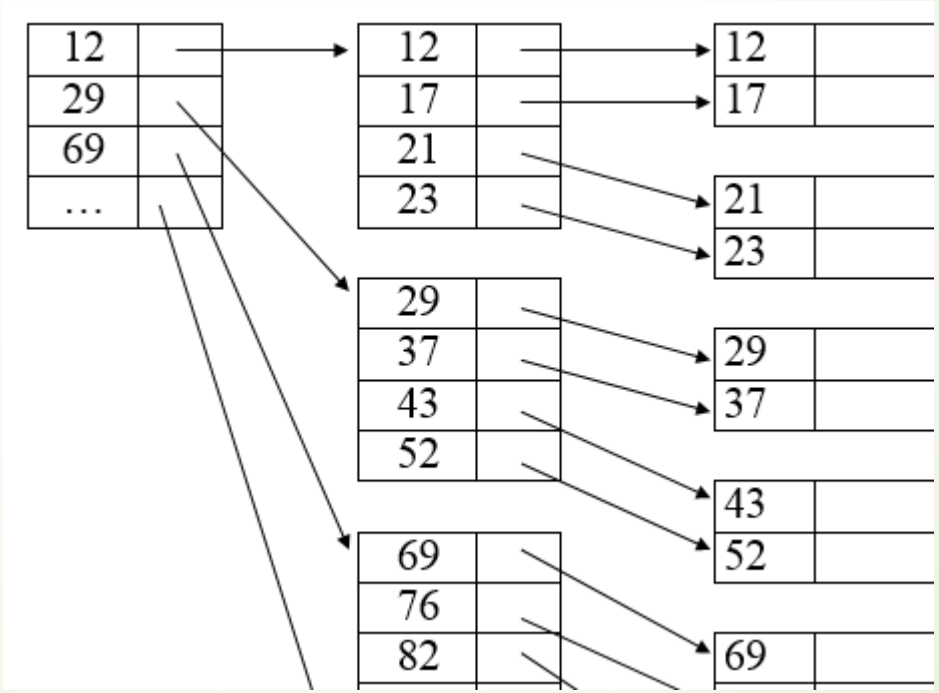
Példa: RITKA indexre



Az indexnek annyi rekordja van, ahány blokkja van az adatállománynak.
Az adatállománynak **rendezetten** kell tárolnia a rekordokat.

Többszintű indexek

Alapötlet: az index rekordjaira is készíthetünk egy újabb indexet.



A második szintű és a további indexek **mindig ritka indexek**

A sűrű index és a ritka index összehasonlítása

Az adatállományon végzett művelet	Sűrű index esetén végzendő művelet	Ritka index esetén végzendő művelet
Túlcsordulási blokk létrehozása	Nincs	Nincs
Túlcsordulási blokk törlése	Nincs	Nincs
Új blokk létrehozása	Nincs	Beszúrás
Blokk törlése	Nincs	Törlés
Rekord beszúrása	Beszúrás	Módosítás[1]
Rekord törlése	Törlés	Módosítás[1]
Rekord szomszédos blokkba mozgatása	Módosítás	Módosítás[1]

(1) - Megjegyzés:
A ritka index esetén csak akkor van szükség módosításra, ha a művelet hatására a blokk első rekordja megváltozik.

Elsődleges, és másodlagos indexek

Elsődleges indexnek nevezzük az olyan indexet, amely meghatározza az adatrekordok fizikai elhelyezését. Pl. egy új rekord beszúrásakor az index egyértelműen meghatározza, hogy hova tehetjük be az új rekordot.

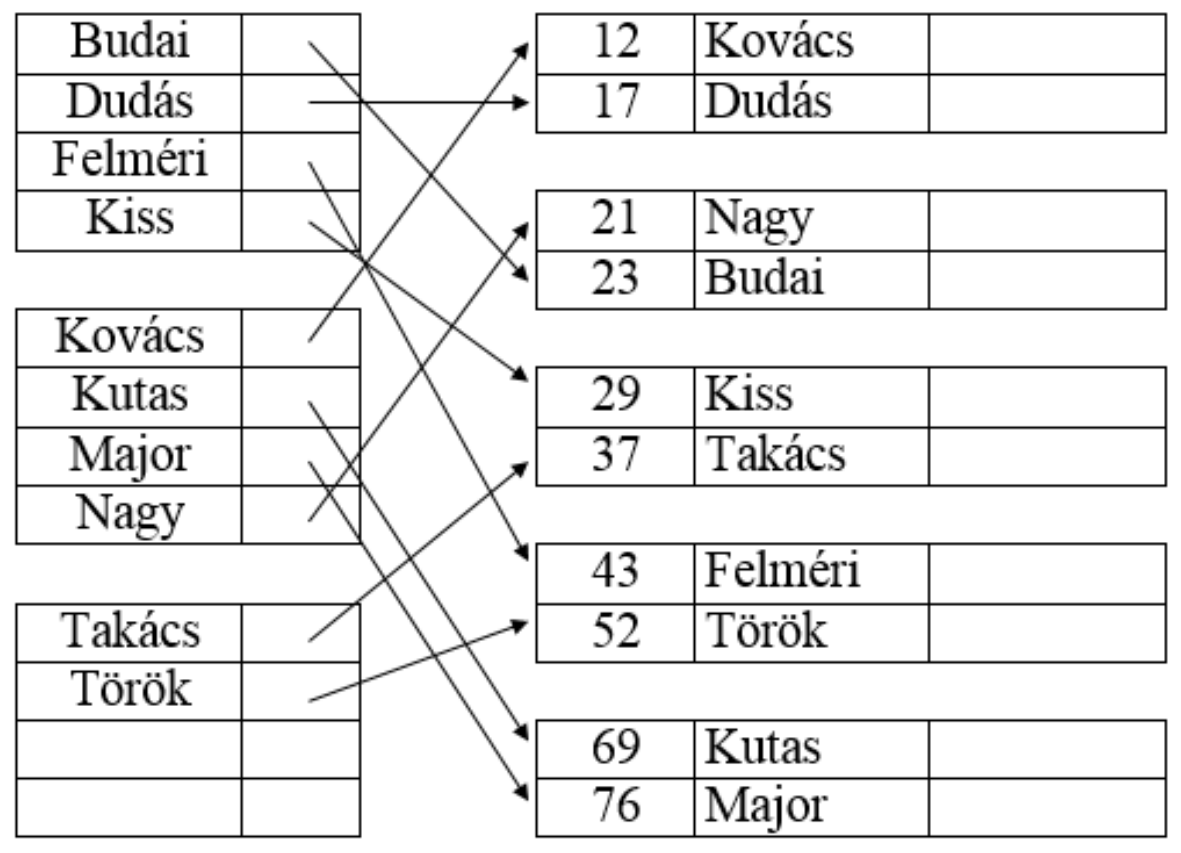
- **A rekordok fizikai elhelyezését az elsődleges indexbeli kulcsértéküket határozza meg!!!**
- **Elsődleges indexből csak egy lehet!!!**

A másodlagos indexet az különbözteti meg az elsődlegestől, hogy az index független a rekordok fizikai elhelyezésétől.

- **Másodlagos indexből többet is létrehozhatunk!!!**

Példa másodlagos indexre

- Hozzunk létre egy másodlagos indexet a második (név) mező alapján



B-fák

A **B-fák** olyan **kiegyensúlyozott keresőfák**, (**mert mindig a középső elemet emeljük ki**) amelyeket úgy terveztek, hogy hatékonyan lehessen alkalmazni őket mágneslemezeken vagy más közvetlen hozzáférésű másodlagos tároló berendezéseken.

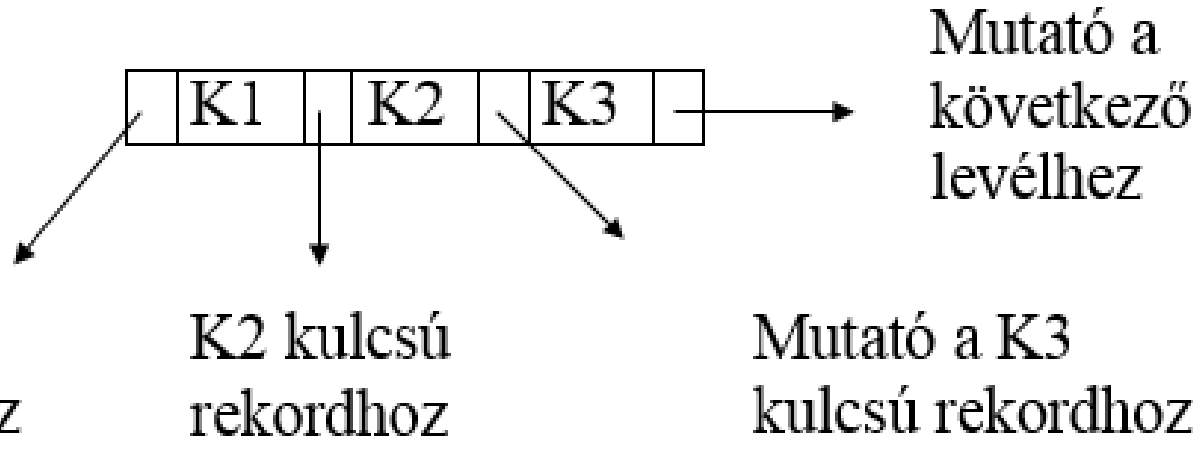
A B-fában a csúcsoknak sok gyerekük lehet. Azaz, egy B-fában az "elágazási tényező" igen nagy lehet, bár erre a felhasznált mágneslemez jellemzői egy felső korlátot adnak.

A B-fa alulról fölfelé építkezik. Jellemzője, hogy a csomópontokat egyenletesen telíti.

Az adatrekordok/kulcsok csak a levelekben vannak tárolva. A belső pontokban a keresést informáló kulcsok és mutatók vannak.
A levelekben balról jobbra nőnek a kulcsok.

B-fák

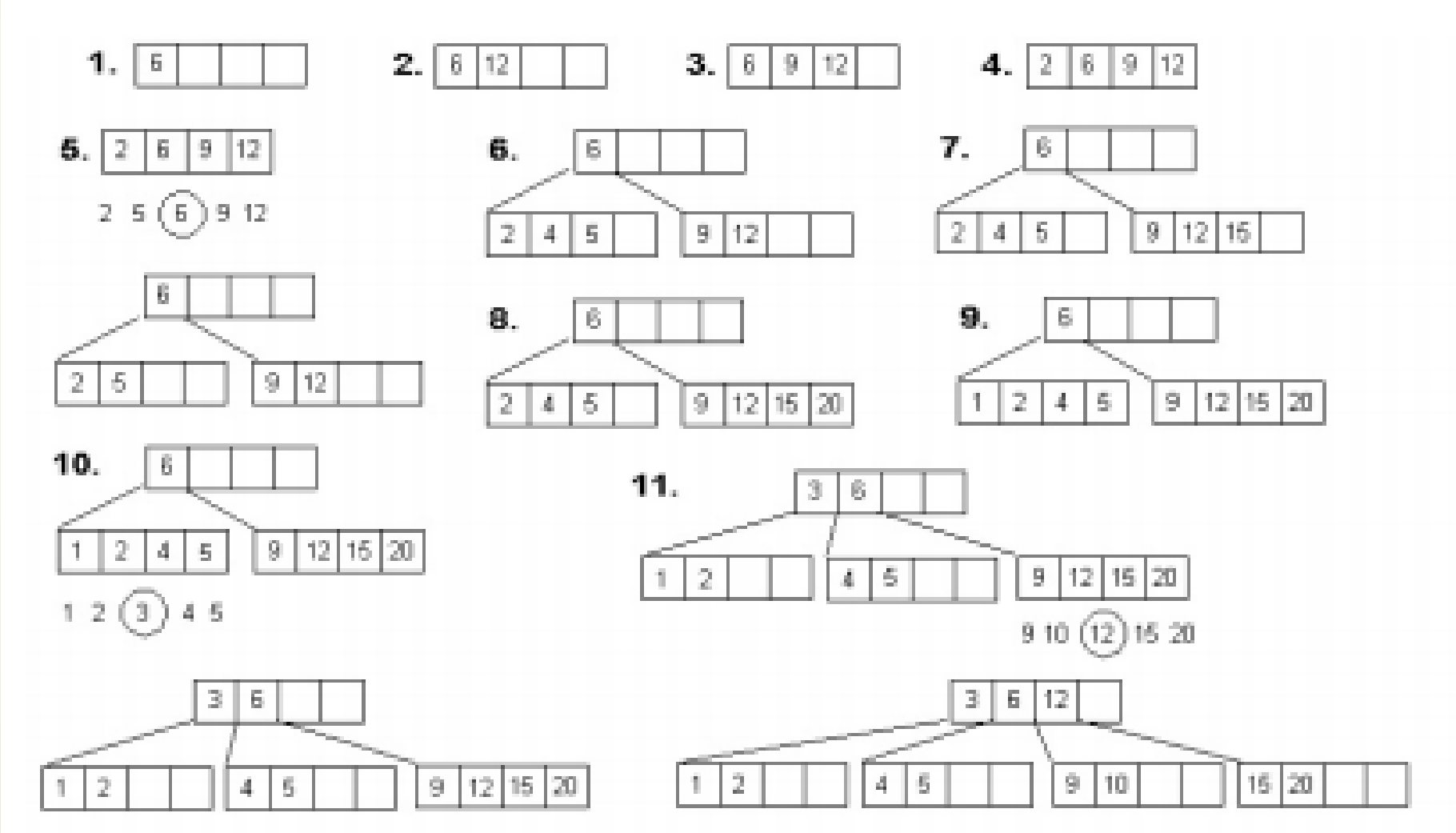
A **B-fa levelei** a következő szerkezetűek:



Egy **n** paraméter meghatározza a **B-fa** blokkjainak az elrendezését.

Minden blokkban **n** *keresési kulcs* és **n + 1** *mutató* van.

B-fa építkezés (4-es blokkokba)
6, 12, 9, 2, 5, 4, 15, 20, 1, 3, 10



B-fa animáció

ANIMÁCIÓ

http://ats.oka.nu/b-tree/b-tree.html

https://visualgo.net/bn/bst