Adatbázis rendszerek I
2011.09.15

Követelmények DBMS-el szemben (nagy adatmennyiség, hatékony védelem, megbízható)

Hogyan valósíthatók meg?
1.Hatékony adathozzáférés
2.szelekció, szűrés
milyen algoritmus?
Alap algoritmus -> B-fa index

Van 1 rekord Lista és kulcs mezőknek megfelelően szeretnék kiválasztani
Algoritmus változatok:
1. lineáris, szekvenciális keresés
Lassú
$O(f) = (g \mid \text{létezik } n_1,n_2 \in N^+ : g(x)<n_1*f(x)+n)$

Az O az egy közelítés a feladat megoldásához szükséges idő és a feladat méretével kapcsán.
A O osztálynál a görbének a jellege számít és lineáris.

Hatékonyságok:
1. O (log x)
2. O (x)
3. O (x^2)
4. O (a^x)

Szekvenciális – O(x)

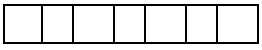
2. Felező keresés
Előfeltétel: rendezett
Minden lépésben felezi a keresési tartományt

Ez lenne jó, miért nem megy a gyakorlatban?
a)A lista nem rendezett. *Miért nem rendezzük?*
-mert nagy költésg
-minden beszúrásánál módosítunk

O(log x) + O(x)

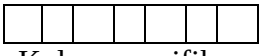
b) Az illesztés költségét láncolt listával meg tudjuk oldani
c) skiplista

A gyakorlatban mit alkalmazunk?
Vegyes megoldás lesz (index struktúra)



Rekord lista, nem rendezett O(1)

Viszont a kereséshez egy segéd rendezett struktúrát alkalmazunk.



Kulcs szpecifikus

Egy rendezett tábla amiben a kulcsok szerepelnek. Minden kulcshoz tartozik egy link.

Index struktúra – kulcsértékek és pointerek kulcs szerint rendezett listája.

Probléma: ha viszont beszúrunk akkor az indexnek is módosulnia kell.
Gyakorlati nyereség a méret különbségből fakad.
Különbség – kis méret estén befér a struktúra a memóriába nagy méret estén nem.

Nyereség:

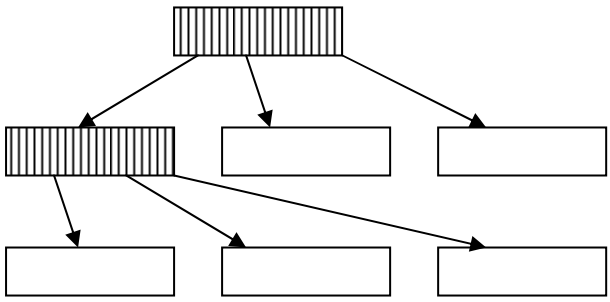
a) olvasás költség
b) rugalmasabb struktúra, mert 1 fizikai sorrendhez több logikai sorrendet is támogat.

Index megvalósítása

Ha az index hosszú akkor lassú lesz
Sok elem esetén az index is kiszorulhat a memóriából.

Hogyan lehet gyorsítani?
Az indexet is indexeljük

Megvalósítása a hierarchikus index-fának
Gyakorlati szempontból a listát darabokba tároljuk.



Fix méretű bejegyzés szám

A B-fa nagy előnye, hogy kiegyensúlyozott.

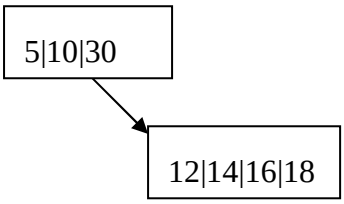
A hagyományos keresőfánál az ágak lefelé növekednek.
A B-fa forgatás nélkül oldja meg.

Beszúrás algoritmusa

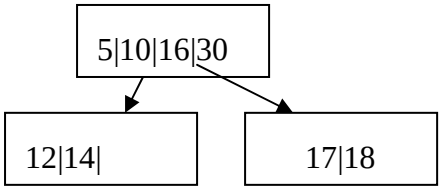
Minden levél azonos szinten van

1. tartalmazó levél megkeresése
2. ha a levél nem telített, elem beszúrása (rendezve)
3. ha nem fér be,
   - logikailag beilleszti az elemet
   - a középső elem kiválasztása
   - új testvér levél létrehozása
   - az elemek felét átviszi
   - a középső elem felkerül a szülőbe
   - a pointerek igazítása

Minta



Vigyük be pl a 17-est



A középső elem beillesztése a szülőbe újabb hasítást válthat ki. Ha megtelik egy gyökér, új gyökér keletkezik.

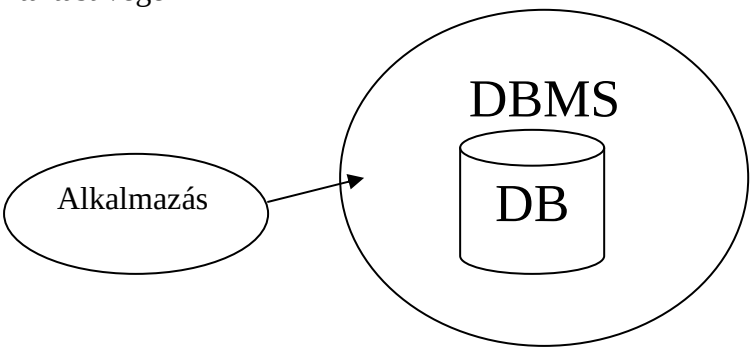
A B+ fánál vannak testvérek közötti pointerek.
A B+ fa akkor gyorsít, ha intervallum keresést végzünk.

Hash – egy alternatívája az indexnek.
Az index összehasonlítással keres, a hash viszont nem.
Időigénye : O(1) – mert lemezolvasásra nincs szükség, tehát a hasító függvény lenne a legjobb megoldás. Problémája a túlcsordulás. Túlcsordulás esetén O(x) – válhat a keresés.
A hasító függvény hátránya:
- nem megbízható
- Akkor használjuk sikerrel, ha a szétosztandó elemek eloszlása ismert. Ha statikus az adat jó a Hash, viszont ha dinamikus akkor az index.

Sok funkció kell a hatékony adatkezeléshez:
- védelem
- értelmezés
- konkurencia
- I/O
- Tranzakció kezelés

DB: Integrált adatrendszer, amely az adatokat adatmodell szerint tárolja, perzisztens adattárolást végez, alapadatok + kapcsolatok + meta adatok

DBMS: olyan programrendszer, amely vezérli a DB-hez való hozzáférést és belső karbantartást végez



Részletes funkcióábra:

Hálózati, kapcsolati réteg
Szintaktikai elemzés
Szemantikai ellenőrzés
védelem
optimalizálás
tranzakció kezelés
Végrehajtó I/O

Ezek biztosítják a centralizált felületet

DBMS használata:

Cél: Üzleti adatok centralizált tárolása

1. Létrehozni a DB adatszerkezetet
2. Használat (lekérdezés, feltöltés)

Tervezés:

1. Szemantikai modell (terv) ER
2. Logikai adatbázis modell (részletes terv)
3. Parancs létrehozás (SQL)
4. futtatás
5. tesztelés

ER modell:

Feladat: rendelés nyilvántartás.
Szemantikai modell célja: információelemek és kapcsolata, struktúrája.
-ER elemei:

- Egyed: önálló léttel bíró összefogó elem (ügyfél) termék
- Kapcsolat: egyedek közötti társítás
- Tulajdonság: Információelemek, melyek az egyedet vagy a kapcsolatot jellemzik.

Kapcsolatok