

# Memóriaadatbázisok

Marton József

marton@db.bme.hu  
BME-TMIT

Adatbázisok elmélete VITMMA13  
2018. február 20.

# Miről lesz szó?

- ▶ Memóriaadatbázisokról általában
  - ▶ Alapelv, kontraszt a „hagyományossal”
  - ▶ Előnyök, erősségek
  - ▶ Hátrányok (?), nehézségek
- ▶ Szervezés
  - ▶ Kommunikáció
  - ▶ ACID D betűje
  - ▶ Fizikai szervezés, lekérdezésvégrehajtás
- ▶ Megvalósítások

## DBMS: a kezdetek

Az 1960-as évek hardverei

- ▶ Egy mag, egyetlen CPU
- ▶ Kevés memória
- ▶ Diszk-alapú adatbázistárolás
- ▶ A diszk lassú

# DBMS: a kezdetek

Az 1960-as évek hardverei

- ▶ Egy mag, egyetlen CPU
- ▶ Kevés memória
- ▶ Diszk-alapú adatbázistárolás
- ▶ A diszk lassú

Válaszok

- ▶ Ütemező algoritmusok
- ▶ Buffer-kezelési algoritmusok
- ▶ Fizikai szervezés: B\*-fa, Heap, vödrös hash
- ▶ A diszk lassú

## DBMS: overhead

Mérés: OLTP-ben a CPU-idő felhasználása

- ▶ buffer pool: 34%
- ▶ latch kezelés: 14%
- ▶ zárkezelés: 16%
- ▶ naplózás: 12%
- ▶ keresési kulcsok összehasonlítása: 16%

## DBMS: overhead

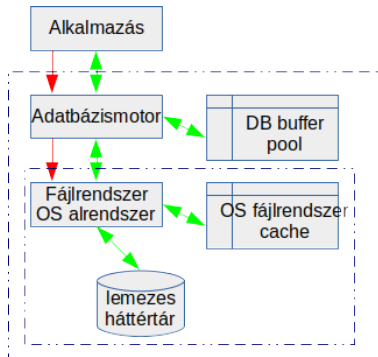
Mérés: OLTP-ben a CPU-idő felhasználása

- ▶ buffer pool: 34%
- ▶ latch kezelés: 14%
- ▶ zárkezelés: 16%
- ▶ naplózás: 12%
- ▶ keresési kulcsok összehasonlítása: 16%
- ▶ érdemi munka: 7%

In: OLTP THROUGH THE LOOKING GLASS, AND WHAT WE FOUND THERE; SIGMOD, pp. 981-992, 2008.

# Adathozzáférés DBMS-ben

- ▶ Az ábrán: DRDB
  - ▶ piros nyilak: üzenetátadás
  - ▶ zöld nyilak: adatáramlás
  - ▶ a szaggatott keret átlépése a futási környezetváltás
- ▶ kliens-szerver modell
- ▶ osztott memória alapú adatelérés
- ▶ durva hozzáférés szabályozás



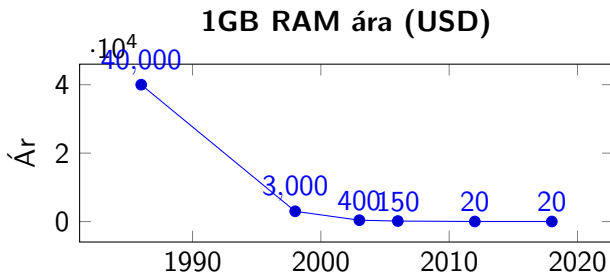
# Memóriaadatbázis: alapelv

## Memóriaadatbázis-rendszer (IMDB) (In-Memory DataBase)

- ▶ az adatok elsődleges példánya a fizikai memóriában
- ▶ DRDB: az elsődleges példány diszken van (Disk-Resident DataBase)  
mondjuk inkább úgy: blokkos tárolón (vö. SSD)



## Az alapelv létjogosultsága



- ▶ Sok és olcsó RAM  
OLTP, OLAP rendszerek számára
- ▶ Opció: adatbázis particionálása IMDB és DRDB részre
  - ▶ automatikusan vagy kézzel konfigurált módon
  - ▶ az adatok migrációja a két rendszer között

## IMDB használatának előnyei

- ▶ gyorsabb adatelérés, tranzakció-feldolgozás akár hard real-time alkalmazások
- ▶ egyes esetekben egyszerűbb alkalmazás-logika diszk-IO alrendszer eliminálása
- ▶ saját adatszerkezetek helyett kész komponens olcsóbb fejlesztés, olcsóbb termék

## IMDB: hátrányok, nehézségek

- ▶ bizonyos esetekben bonyolultabb alkalmazás-logika
  - ▶ tartósság biztosítása
  - ▶ biztonsági mentések szervezése
- ▶ a rendszer indulásakor fel kell tölteni az adatbázist
  - ▶ mentésből, fejállomásról letöltve vagy on-line adatgyűjtésből
  - ▶ segédstruktúrák online felépíthetők
- ▶ Megéri-e: előnyök vs. hátrányok: a konkrét feladat határozza meg

# IMDB: megvalósítási kihívások

- ▶ optimalizált adatszerkezetek
  - ▶ a fizikai memória véletlen elérését kihasználják
  - ▶ zárkezelés, tranzakciók ütemezése
  - ▶ megfelelő granularitású zárok
- ▶ tartósság
  - ▶ HW támogatás: elemes RAM, hálózaton szinkronizáló RAM-kártyák
  - ▶ memrisztor (RRAM: Resistive random-access memory)
  - ▶ szinkron vagy aszinkron naplózás
  - ▶ nem követelmény

## Vajon IMDB-nek minősül?

- ▶ egyedi alkalmazás saját adatszerkezetekkel
- ▶ hagyományos DBMS úgy, hogy
  - ▶ nagy fájlrendszer cache az operációs rendszerben
  - ▶ nagy buffer pool
  - ▶ adatfájlok RAM-diszken

# Tranzakciós tulajdonságok I.

Szemben a DRDB-nél megismertekkel, az IMDB rendszerekben:

- ▶ A – atomicitás
  - ▶ nincs markáns különbség
  - ▶ gyorsabb tranzakciók, magasabb zár-granularitás
- ▶ C – konzisztencia
  - ▶ nincs markáns különbség
- ▶ I – izoláció
  - ▶ sorosítható helyett valódi soros ütemezés
  - ▶ környezetváltások: a processzor-cache, mint környezet
- ▶ D – tartósság

## Tranzakciós tulajdonságok II. A tartósság

Az adatbázisba „írt” adatok „megmaradnak”

- ▶ diszk: passzív
- ▶ memória: aktív
  - ▶ tápfeszültség kimaradásakor törlődik
  - ▶ elemmel támogatott RAM
- ▶ bithiba – modul meghibásodások
  - ▶ ECC (vö: diszkek: RAID)
  - ▶ modultöbbszörözés
- ▶ naplózás (és persze mentések)

## Tranzakciós tulajdonságok III. A naplózás

- ▶ ... de hova?
  - ▶ diszkre: már megint diszkIO
  - ▶ stable memory
    - kicsi, nagy megbízhatóságú memória-megoldás
  - ▶ memrisztor
    - amikor elérhető lesz, de még drága
- ▶ ... de mikor?
  - ▶ írási műveletkor azonnal
  - ▶ tranzakció kommitjakor
  - ▶ még később?!
  - ▶ ún. durable commit (Oracle TimesTen)



## Tranzakciós tulajdonságok III. Durable commit

**Table:** durable commit hatása a tranzakciófeldolgozási kapacitásra

select/insert/update	70/15/15	40/30/30	20/40/40
tartós kommit	100%	52%	38%
nem tartós kommit	918%	714%	626%

- ▶ A mérés a tptBm benchmark eredményeit mutatja
- ▶ Mérési környezet: AMD Athlon64 3000+ (1809 MHz) CPU, GiB RAM, 7200 rpm fordulátú Seagate Barracuda UltraATA 100-as diszk, Debian Linux 3.1 amd64 OS, TimesTen 6.0.2-es, 64 bit

## Index adatszerkezetek I. Hash és B\*-fa

- ▶ Hash-alapú segédstruktúrák
  - ▶ kiterjeszthető hash
  - ▶ többszintes hash
- ▶ B\*-fa (IMDB-hez lassú?)

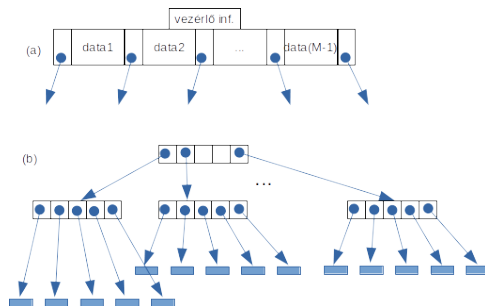


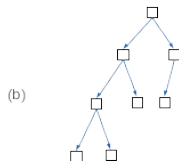
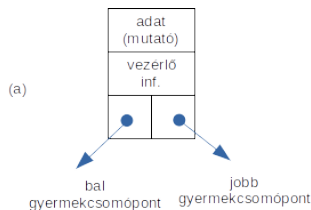
Figure: B\*-fa: (a) egy B-fa csomópont, (b) egy B-fa

## Index adatszerkezetek II. Az AVL-fa

- ▶ AVL-fa
  - ▶ bináris keresőfa  
AVL-tulajdonsággal
  - ▶ gyors navigáció
  - ▶ gyenge memóriakihasználás
- ▶ Az ábrán:

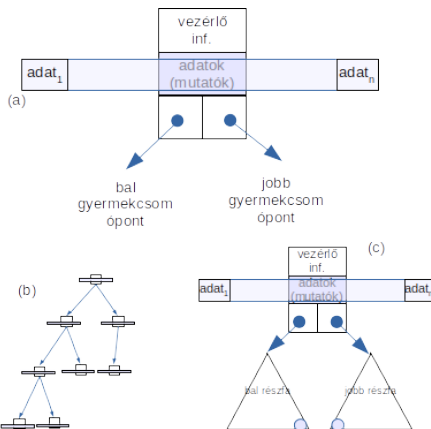
(a) egy AVL-fa csomópont

(b) egy AVL-fa



## Index adatszerkezetek III. Az T-fa

- ▶ B\*-fa és AVL-fa előnyei összegyűrva
  - ▶ gyors navigáció
  - ▶ jó memóriakihasználás
- ▶ Az adatok tárolása
  - ▶ adat maga
  - ▶ mutató
  - ▶  $m + \text{adat rész}$  (pkT-fa)
- ▶ Az ábrán:
  - (a) egy csomópont
  - (b) egy T-fa
  - (c) elemek helye a fában



# Relációk tárolása

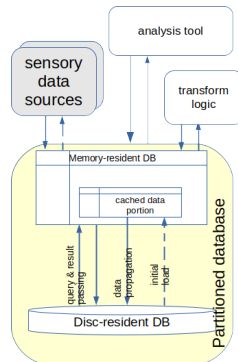
- ▶ rekordok vagy mutatók tömbje
- ▶ idegen kulcsok követésének (illesztés) támogatása
  - ▶ avagy „előreszámított (fél)illesztések”
  - ▶ DRDB: klaszterezés
  - ▶ IMDB: a kulcs értéke helyett mutató

# Lekérdezésvégrehajtás

- ▶ Vetítés, szelekció
  - sokszor olcsóbb az eredményreláció leírójának módosítása és on-the-fly számítás, mint a tényleges számítás
- ▶ Illesztések
  - l. a mutatók követéséről írottakat
- ▶ Költség
  - ▶ DRDB: tipikusan az eredmény kiírásának költségét nem számoljuk (miért?), de
  - ▶ IMDB: a saját címtér miatt nem mindig merül fel

## Architektúra: partícionálás

- ▶ ha a teljes adatbázis IMDB-ben nem
  - ▶ megoldható v.
  - ▶ racionális
- ▶ Hibrid megoldás
  - ▶ tiszta IMDB
  - ▶ in-memory cache
  - ▶ DRDB



Az ábrán: az elemzőeszközök a partícionált adatbázishoz csatlakoznak, amíg az adatforrások friss adatai közvetlenül az IMDB részbe érkeznek, hogy minél hatékonyabb lehessen az előfeldolgozásuk.

# Ha nincs IMDB I.

- ▶ RAM-diszk
- ▶ HW-tuning
- ▶ memóriatuning: buffer-pool kezelési policy
  - ▶ Keep
  - ▶ recycle
  - ▶ standard



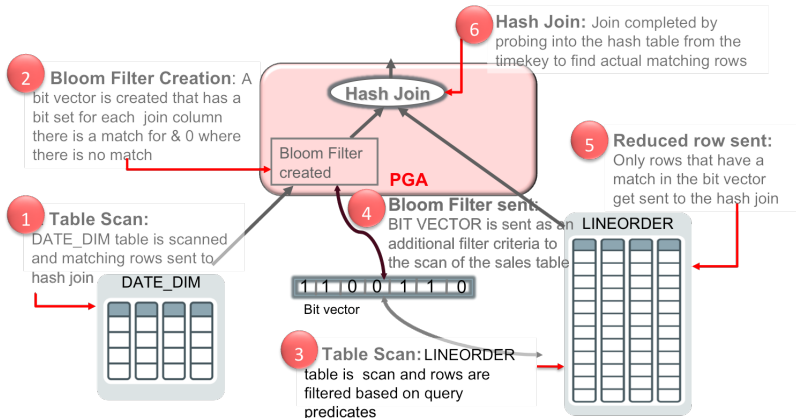
## Ha nincs IMDB II. Oracle Database In-Memory

- ▶ oszloporientált
- ▶ tömörítés, kibontás nélküli keresés
- ▶ IMCU indexelés, szótár
- ▶ SIMD támogatás
- ▶ hash-join Bloom filter segítségével

További olvasnivaló: Oracle blog: Getting started w/ Database In-Memory:

- ▶ Elméleti jellegűek: Part IV: joinok, Part V: aggregáció
- ▶ Üzemeltetés/technológiai jellegűek: Part I: bekapcsolás, Part II: betöltés, Part III: lekérdezések, Part VI: lekérdezésenkénti be/kikapcsolás

# Bloom filter illusztráció



Ábra forrása: Oracle blog: Database In-Memory, Part IV - Joins

# Megvalósítások I.

- ▶ Peloton
- ▶ Oracle TimesTen
- ▶ SAP Hana
- ▶ MySQL memory engine

## Megvalósítások II. Peloton

- ▶ önmenedzselő
- ▶ MI alapú optimalizáció
- ▶ non-volatile memória támogatás
- ▶ zármentes MVCC
- ▶ latch-mentes Bw-fa indexek
- ▶ PostgreSQL protokol-kompatibilitás

További infó: [pelotondb.io](http://pelotondb.io)