Levytiedoston käsittely

- Olkoon
  - S= kohdistusaika
  - R= pyörähdysviive
  - T= jakson siirtoaika= kierrosaika/uran lohkojen lukumäärä.
- Lohkon saantiaika = S+R+T
  - kohdistusaika ja pyörähdysviive vaihtelevat, arvioissa käytetään usein keskimääräistä hakuaikaa
- Sivupyynnön toteutuksen palveluaikaan on vielä lisättävä jonotusaika sillä levyohjain voi suorittaa vain yhtä pyyntöä kerrallaan – jonotusaikaan vaikuttaa kaikki levylle kohdistuva kuorma

Levytiedoston käsittely

- Haettaesa k lohkoa, lohkojen yhteenlaskettu saantiaika on
  - enintään k*(S+R+T), jos lohkot sijaitsevat satunnaisesti levyllä
  - S+k*(R+T), jos lohkot sijaitsevat satunnaisesti samalla sylinterillä ja
  - minimissään S+ R + k*T, jos lohkot ovat peräkkäin samalla sylinterillä
- Kaavojen tekijöistä S ja R ovat samaa millisekunneissa mitattavaa kokoluokkaa ja T noin tuhannesosa niistä (mikrosekunteja)
- Jos S=5ms, R=5ms ja T=0.02 ms ja k=1000, niin kokonaissaantiajan vaihteluväli olisi 0.030s - 10.02s

Levytiedoston käsittely

- Tiedostot sijoitetaan levylle
  - yhdelle tai useammalle peräkkäisistä lohkoista koostuvalle alueelle
    - voidaan hyödyntää peräkkäiskäsittelyn nopeutta koko tiedoston lukemisessa
    - puskurien hallinnassa voidaan käyttää ns. ennakoivaa hakua (pre-fetch) – haetaan valmiiksi puskureisin järjestyksessä seuraavia sivuja vaikka niitä ei ole vielä pyydetty

Levytiedoston käsittely

- Vaikka tiedoston tietueet alunperin sijoitettaisiin tiiviisti lohkoihin voivat poistot ja muutokset huonontaa rakennetta
  - peräkkäisjärjestykseen jää väliin tyhjiä lohkoja poistojen takia
  - jatkotietueet ylivuotolohkoissa vaativat poikkeamista optimaalisesta lukemisjärjestyksestä – mahdollisesti jopa siirtymistä pois sylinteriltä

RAID–levyt ja rinnakkaisuus

- RAID = redundant array of independent disks
- ryhmä toisistaan riippumattomia levyjä, jotka toimivat rinnakkain yhtenä loogisena levynä
- tavoitteena luotettavuuden ja tehokkuuden parantaminen
  - tehokkuutta parannetaan viipaloinnilla (striping)
  - luotettavuutta redundanssilla (ylimäärällä)

RAID–levyt ja rinnakkaisuus

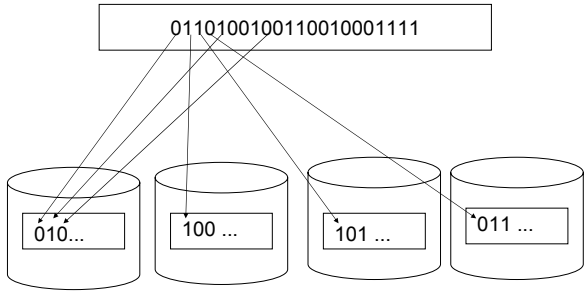
- Viipaloinnin periaatteena on jakaa tiedosto usealle levylle. Tiedonsiirtoa levyjen ja puskureiden välillä voidaan suorittaa rinnakkain koska jokainen levyohjain toimii itsenäisesti. Näin tiedonsiirtonopeus kasvaa.
- Käytössä on eri viipalekokoihin (striping unit) perustuvia viipalointiperiaatteita. Jos käytettävissä on D levyä, sijoitetaan i:s viipale levylle i mod D.

RAID–levyt ja rinnakkaisuus

- Viipalekoko voisi olla yksi bitti.
  - Tällöin D peräkkäistä bittiä ovat kukin eri levyillä ja kaikki D levyä osallistuvat jokaiseen hakuoperaatioon.
  - Koska pienin siirtoyksikkö on lohko joudutaan sivupyynnön yhteydessä siirtämään D lohkoa.
    - Jos käsittely on peräkkäiskäsittelyä, on siirron yhteydessä tehty ennakoiva haku D-kertaisella siirtonopeudella
    - Jos käsittely on hajakäsittelyä, on siirretty turhaa dataa, mutta siihen ei ole mennyt enempää aikaa kuin yhtä levyä käytettäessä

RAID–levyt ja rinnakkaisuus

yhden bitin viipalekoko – 4 levyä



RAID–levyt ja rinnakkaisuus

- Lohkotason viipaloinnissa viipaleen koko on yksi lohko.
  - Jos sivupyynnöt kohdistuvat eri levyille voidaan pyynnöt suorittaa rinnakkain. Jonotusaika pienenee joten myös hajakäsittely nopeutuu.
  - Peräkkäiskäsittelyssä peräkkäiset lohkot sijaitsevat eri levyillä, joten tiedonsiirtonopeus kasvaa
  - Suurella levymäärällä saadaan suuremmat tehot

RAID–levyt ja rinnakkaisuus

- Tietojen jakaminen usealle levylle aiheuttaa luotettavuusongelman.
- N levyn järjestelmässä häiriön todennäköisyys on N-kertainen yhteen levyyn verrattuna
- Jos yhden levyn kohdalla häiriövälin odotusarvo on 50000 tuntia (5.7 vuotta), niin 100 levyn järjestelmässä se onkin enää 50000/100 =500 tuntia eli noin 21 päivää
- Luotettavuutta lisätään ylimäärällä, jolloin häiriöitä voi sietää.

RAID–levyt ja rinnakkaisuus

- Eräs ylimäärän muoto on peilaus (mirroring)
  - Tieto kirjoitetaan kahdelle identtiselle levylle.
  - Jos toinen levy vikaantuu, voidaan jatkaa toisella kunnes vika saadaan korjattua.
  - Kumpaakin levyä voi käyttää sivupyyntöjen toteutukseen, pyyntö ohjataan levylle, jolla on lyhyempi saantiaika.
  - Levyjen tehollinen kapasiteetti puolittuu
  - Kallis ratkaisu

RAID–levyt ja rinnakkaisuus

- Peilauksen vaihtoehtona on erilaisia virheenkorjauksen mahdollistavia tekniikoita, Hamming koodi, bittitason tai lohkotason viipalointiin perustuvat pariteettibitit omalla levyllään tai hajautettuna yksikön levyille.

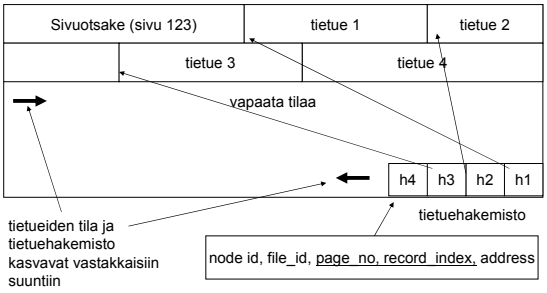
Tietojen tallennusrakenteet

- Jokaisella tiedostolla on otsake (header), joka sisältää tiedostoon liittyvää hallintatietoa
  - tiedot tiedostoon kuuluvista lohkoista esim. taulukkona, joka voi muodostua ketjutetuista osista
  - tietoja tietueiden rakenteesta
  - muuta hallintatietoa

Tietojen tallennusrakenteet

- Jokaiseen sivuun sisältyy sivuotsake (page header), johon sisältyy
  - sivutunniste (page identifier
    - yleensä sivun järjestysnumero tiedostossa, on tiedostokuvaajan perusteella muunnettavissa fyysiseksi levyosoitteeksi
  - ensimmäisen ja viimeisen tietueen osoite
  - käyttötietoa
- Sivuun sisältyy myös tietuehakemisto, josta saadaan kunkin tietueen osoite
- Lisäksi sivulla ovat tietueet

Sivun rakenne



Sivun rakenne

- Tietueen tunnisteen (record identifier, RID) muodostaa pari
  - sivutunniste
  - tietueen indeksi (tietuehakemistossa)
- Tuple identifier (TID) = record identifier

Tietueen rakenne

- Samassa tiedostossa olevat tietueet voivat olla keskenään
  - saman pituisia, eli kiinteäpituisia (fixed length)
  - vaihtuvapituisia (variable length)
    - erityyppisiä esim. kurssitietueita ja osallistujatietueita
    - saman tyyppisiä, mutta tietokenttien pituus tai määrä vaihtelee
- Tietueessa voi olla kiinteäpituisia tai vaihtuvapituisia kenttiä, yksikin vaihtuvapituinen kenttä tekee tietueesta vaihtuvapituisen

Tietueen rakenne

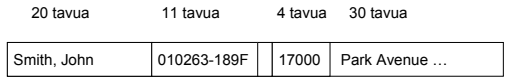
- Kiinteäpituinen kenttä on jokaisessa tietueessa saman pituinen
  - smallint, integer, float, double, date ja timestamp tyyppiset arvot tallennetaan tyypillisesti (ei kuitenkaan välttämättä) kiinteäpituisiin kenttiin – kentän pituus riippuu tietotyypistä. Esimerkiksi kokonaisluvut voitaisiin tallentaa binäärisinä 4 tavun pituisina kenttinä
  - SQL:n tietotyyppi char määrittelee kiinteänpituisen merkkijonon. Sillekin kiinteäpituinen kenttä soveltuisi

Tietueen rakenne

- Vaihtuvapituisia kenttiä käytetään tyypillisesti tilanteissa, joissa tietoalkioiden arvot vaihtelevat merkittävästi pituudeltaan
  - Esimerkiksi vaihtuvapituiset merkkijonot
  - Yleensä vaihtuvapituisilla kentillä pyritään säästämään tilaa, mutta tilansäästöä voidaan saada muutenkin esimerkiksi tiivistämällä tietueet

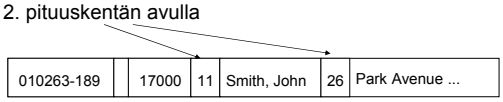
Tietueen rakenne

- Relaation employee(name, ssn, salary, address) monikon
  ('Smith, John', '010263-189F', 17000, 'Park Avenue ...') esittäminen:
- kiinteänmittaisilla kentillä:



Tietueen rakenne

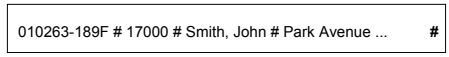
- Vaihtuvanmittaiselle kentälle erilaisia esitystapoja:
- 2. pituuskentän avulla



- kiinteänmittaiset alussa kiinteissä paikoissa
- vaihtuvanmittaisen osan alkukohta samoin kenttien järjestys (vähintään) on kuvattava esim. sivun otsikkotietueessa

Tietueen rakenne

- 3. erotinmerkkien avulla



Kenttien järjestys kiinnitetty

Tietueen rakenne

- Oraclen tietuerakenteessa kaikki kentät ovat periaatteessa vaihtuvapituisia/valinnaisia
  - Kentät esiintyvät tietueessa siinä järjestyksessä, missä ne on esitelty 'create table' -lauseessa (myöhemmät lisäykset tietueen loppuun).
  - Kunkin kentän alussa on kentän järjestysnumero ja arvon pituus
  - Jos kenttä on tyhjä (null), niin se puuttuu kokonaan tietueesta.