Osnove Podatkovnih Baz

Predavanja

INFORMACIJE

Literatura
- Ramakrishan, R., Gehke, J.: Database management Systems, 3rd Edition, McGraw-Hill, 2004
- Prosojnice
- Korth, F.H., Silberschatz, A., Sudarshan, S.: Database Systems Concepts, 6th Edition, McGraw-Hill, 2010

Ocena:
- Pisni izpit = 40%
- Domače naloge = 40%
- Ustni izpit = 20%

7.10.24

UVOD

- SUPB - sistem za upravljanje podatkovnih baz
- Info. sistem temelji na relacijah - relacijski sistem
- Podatkovni modeli:
  - Hierarhični - starejši, nadomeščen z relacijskim,
  - Relacijski - uporaben za različne sisteme,
  - Objektni, ...
- SUPB - podatki v tabelah,

Datoteke vs. SUPB

- Aplikacije prenašajo velike količine podatkov med dinamičnim spominom in diskom
- Posebna koda za posamezne poizvedbe
- Zaščita podatkov pred nekonsistenco, ki je lahko posledica večih hkratnih uporabnikov
- Zaščita pred izpadom sistema
- Varnost in kontrola dostopa

Zakaj SUPB?

- Aplikacija mora brati/pisati velike količine podatkov iz diska
  - Vmesni pomnilnik, bloki, učinkovit dostop do podatkov
  - Podatkovna neodvisnost
- Zmanjšan čas razvoja aplikacije
- Podatkovna integriteta in varnost
- Uniformno administriranje podatkov
- Hkraten dostop, transakcije zaščita pred sist napakami

Podatkovni modeli
- Podatkovni model je zbirka konceptualnih gradnikov (jezik) za opis podatkov
- Sheme
- Relacijski podatkovni model

Abstrakcija
- Več pogledov, ena konceptualna shema in fizična shema
- Primer:
  Konceptualna shema:
  - Študenti (sid: string, ime: string, login: string...)
  - Predmeti (pid: string, pime: string, točke: int)
  - Vpis (sid: string, pid: string, ocena: string)
  Fizična shema
  - Relacije shranjene v neurejenih datotekah
  - Index je def. na 1. stolpcu relacije Študenti
  Zunanja shema (Pogled)
  - Predmet_Info (pid: string, vpisani: int)

Podatkovna neodvisnost
- Aplikacije se neukvarjajo s tem kako so podatki shranjeni
- Logična PN: Zaščita pred spremembami v logični strukturi
- Fizična PN: Zaščita pred spremembami fizične strukture

Kontrola sočasnega dostopa
- Sočasno izvajanje uporabniških programov je bistveno za dobre performanse SUPB

Transakcija
- Osnovni koncept, ki je atomarna sekvenca akcij SUPB
- Vsaka transakcija, ki je izvrši v celoti, mora pustiti PB v konsistentnem stanju, če je PB konsistentna

Razvrščanje sočasnih transakcij
- SUPB zagotavlja izvajanje v zaporedju
- Smrtni objem (Dead Lock!)

Zagotavljanje atomičnosti
- SUPB zagotovi atomičnost četudi se zgodi sistemska napaka
- WAL sistem - Write ahead logging

Dnevnik (log)
- Shrani vse spremembe v PB

Struktura SUPB
- Tipična nivojska arhitektura
- Nivoji morajo omogočati kontrolo hkratnega dostopa in reševanje

RELACIJSKI PODATKOVNI MODEL

Zakaj študij relacijskega modela?

- Najbolj široko uporabljen PM
  - Relacije imajo močne matematične osnove
  - System R, IBM, 1974 (first implementation of SQL)
  - 80% vseh SUPB je relacijskih
- 1980-1995: vzpon objektno-usmerjenih sistemov
  - ObjectStore, ObjectDatabase++, GemStone/S, ...
  - Samo peščica OO SUPB na trgu leta 2000
- 1995-2000: Objektno-relacijski model
  - Relacije objektov so preslikane v ravninski relacijski model
  - Implementacije: Oracle, DB2, Sybase ...
- 2000-2010: NoSQL gibanje
  - Kaj se je zgodilo?
    - ↳ Masovni podatki, internetni inform. sistemi, nestrukturirani podatki, semi-strukturirani podatki, raziskovalni podatki, multimedija ...
    - ↳ Obstoječi SUPB ni majo primerne skalabilnosti in dostopnosti za delo z masovnimi podatki
  - Istočasno:
    - ↳ Nove tehnologije: obsežen RAM, SSD diski, masovni HDDs, vzporedne in porazdeljene arhitekture, več procesorski sistemi, več jedrni procesorji, „samostojni" sistemi, ...
  - Rezultat:
    - ↳ Vzpon NoSQL sistemov

- Obstoječi NoSQL sistemi:
  - MongoDB, CouchDB, Dynamo, ...
- 2010-2020: NewSQL sistemi
  - Razvita tehnologija se uporabi v novih relacijskih SUPB!
  - Na novo razviti SUPB
    - Google: Megastore, Spanner, F1
    - Amazon: RDS

Relacijska PB
- Relacijska PB: množica relacij
- Relacija: 2 dela
  - Instanca: tabela, ki ima vrstice & stolpce
    #vrstic = kardinalnost, #stolpcev = stopnja
  - Shema: določča ime relacije ter imena in tipe vseh stolpc

Relacijski poizpraševalni jeziki
- Največja moč relacijskega modela
  - SQL enostaven poizpraševalni jezik
- Uprašanja se lahko pišejo intuitivno *
  - Razlog: naravna semantika
  - Optimizator pogosto prevede operacije (logično ekvivalentna poizvedba, ki vrne isti rezultat)

Poizpraševalni jezik SQL
- Razvit pri IBM leta 1970
- Potreba po standardu

Kreiranje relacij v SQL
- Z besedo CREATE, specificiramo tipe za vse stolpce

Brisanje in spreminjanje relacij
- Brišemo z besedo DROP
- Spreminjamo z ALTER

Dodajanje in brisanje zapisov
- Ustavljanje zapisa z INSERT INTO
- Brišemo zapise z DELETE

Integritetne omejitve (IO)
- IO: pogoj, ki mora biti izpolnjen za vsako n-terico relacije
  *

Primarni ključ
- Množica atributov je ključ če:
  1. Ne obstajata dva enaka zapisa
  2. To neveljа za nobeno podmnožico
- Če 2. ne velja, imamo super ključ
- Lahko obstaja več ključev na za relacijo, primarnega izberemo

Primarni in kandidatni ključ v SQL
- Običajno je navolјo več kandidatnih ključev, izmed njih izberemo enega primarnega
- Ob nepravilni uporabi IO lahko onemogočimo vnos dejanskih podatkov iz realnega sveta

Tuji ključi in referenčna integriteta
- Tuji ključ: Mn. atrib. neke relacije, ki referencira zapise druge relacije

Zagotavljanje referenčne integritete
- Če brišemo iz ene tabele, nato izbrišemo iz druge tabele
- Zavrnemo brisanje iz tabel

Od kod so prišle IO omejitve?
- IO so osnovane na pomenu okolja, ki ga modeliramo
-

RELACIJSKA ALGEBRA

Relacijski poizpraševalni jezik
- Poizpraševalni jezik: Omogoča urejanje podatkov in poizvedovanje po podatkih v PB
- Relacijski model podpira enostavne PJ z veliko izrazno močjo
- Poizpraševalni jezik != programski jezik

Formalni relacijski PJ
- Dva formalna (matematična) jezika tvorita osnovo za „realne" PJ (npr. SQL)
  - Relacijska algebra
  - Relacijski račun

Osnove

- Poizvedba je izvršena nad instancami relacij in rezultat poizvedbe je instanca neke relacije
  - Shema vhodnih relacij - fiksna
  - Shema rezultata - fiksna → določena s pravili gradniki TJ
- Notacija osnovana na poziciji oz. imenih atributov

Relacijska algebra

- Osnovne operacije
  - Selekcija (δ)
  - Projekcija (π)
  - Produkt ($\times$)
  - Razlika ($-$)
  - Unija ($\cup$)
- Dodatne operacije
  - Presek, stik, deljenje, preimenovanje
  - Niso nujne, so pa zelo koristne
- Vsaka operacija vrne relacijo kot rezultat
  - Operacije se lahko sestavljajo - funkcijski jezik

Projekcija

- Izbere atribute v listi projekcije iz relacije
- Shema rezultata vsebuje samo atribute, ki so v listi projekcije z istimi imeni kot v vhodni relaciji
- Duplikati? Realni sistemi tipično ne odstranijo duplikatov, če uporabnik tega ne zahteva

Selekcija

- Izbere vrstice, ki ustrezajo pogoju
- Ni duplikatov v rezultatu
- Shema rezultata identična shemi vhodnih relacij
- Rezultat je lahko vhodna relacija drugi operaciji

Unija, presek, razlika
- Vse operacije so binarne in vhodni relaciji morata biti unija-kompatibilni
  - Enako št. atributov
  - „Pripadajoča" polja imajo enake tipe

Produkt
- Kartezijski produkt: vsaka vrstica S1 se poveže z vsako vrstico R1
- Shema rezultata ima po en atribut za vsak atribut relacij
- Preimenovanje

Stik (Join)
- S pogojem:

$$R \bowtie_C R = \sigma_C(R \times S)$$

- Shema rezultata: enaka kart. prod.
- Manj kriterijev kot produkt (izračuna hitreje
- Theta-stik

Stiki
- Equi-stik: Poseben primer, izvede stik po skupnih atributih
- Shema rezultata: podobno kartezijskemu prod., samo ena vrednost enačenih atributov je v rezultatu
- Naravni stik: Equi-stik po vseh skupnih atributih

Deljenje

- Ni osnovna operacija, uporabna za izražanje vprašanj
  - Poišči mornarje, ki so rezervirali vse ladje
- Naj ima A dva atributa x in y, B pa samo atribut y
  - $A/B = \{\langle x \rangle \mid \exists \langle x, y \rangle \in A \ \forall \langle y \rangle \in B\}$
  - A/B vsebuje vse n-terice x (mornarji) tako, da za vsako n-terico y (ladja) v B, obstaja n-terica xy v A
- V splošnem sta x in y lahko poljubna seznama atributov; y je seznam atributov v B, in $x \cup y$ je seznam atributov v A.

Izražanje A/B z osnovnimi operacijami

- Deljenje ni nujno potrebna operacija; uporabna bljižnica
- Ideja: A/B = izračunaj vse vrednosti x, ki niso izločene z vrednostjo y v B

Ekvivalence operacij RA

- Selekcija: $\delta_{c_1 \wedge \ldots \wedge c_n}(R) = \delta_{c_1}(\ldots(\delta_{c_n}(R)\ldots))$ - Razcep
  $\delta_{c_1}(\delta_{c_2}(R)) = \delta_{c_2}(\delta_{c_1}(R))$ - Komutativnost
- Projekcija: $\pi_{a_n}(R) = \pi_{a_n}(\ldots(\pi_{a_m}(R)\ldots)$ - Razcep
- Stik: $R \bowtie (S \bowtie T) = (R \bowtie S) \bowtie T$ - Asociativnost
  $R \bowtie S = S \bowtie R$ - Komutativnost

⇒ Dokaži $R \bowtie (S \bowtie T) = (T \bowtie R) \bowtie S$

- Spuščanje selekcije/projekcije proti listom
  $\delta(R \bowtie S) = \delta(R) \bowtie S$, če selekcija izbira samo atribute R
  $\pi(R \bowtie) = \pi(R) \bowtie S$, če projekcija uporabi samo atribute R

RELACIJSKI RAČUN

Relacijski račun
- Dva jezika: n-terični RR (TRR) in domenski RR (DRR)
- izrazi uebujejo spr., konst., primerjalne operacije, logične operacije in kvantifikatorje
  - TRR: spremenljivke omejene na n-terice
  - DRR: -||- na domene atributov
  - TRR in DRR so podmnožice predikatnega računa
- Izraze imenujemo formule

N-terični RR
- Poizvedba ima obliko $\{T \mid p(T)\}$, kjer T edina prosta spr.
- Rezultat vsebuje n-terice T, za katere p(t) vrne true
- TRR je rekurzivno def.

TRR formule
- Atomarni izrazi:
  $R \in R_{name}$ ali R.a op S.b ali R.a op constant
  op = $\{<, >, \leq, \geq, =, \neq\}$
- Formule:
  Atomarni izraz, ali
  $\neg p, p \wedge q, p \vee q$ – p, q izraza, ali
  $\exists R(p(R))$, kjer n-terica R prosta v p(R), ali
  $\forall R(p(R))$, kje n-terica R prosta v p(R)
- Uporaba kvantifikatorjev poveznje s spremenljivko R

Domenski RR
- Uprašanje ima obliko: $\{\langle x_1,\dots,x_n\rangle \mid p(x_1,\dots,x_n)\}$
- Odgovor sebaje n-terice $\langle x_1,\dots,x_n\rangle$ za katere vrne izraz $p(\langle x_1,\dots,x_n\rangle)$ vrednost true.
- Formula rekurzivno def.

DRR formule
- Atomarna formula
  $\langle x_1,\dots,x_n\rangle \in R_{name}$, ali X op Y ali X op constant
  $op = \{<, >, \leq, \geq, =, \neq\}$
- Formula
  - atomična formula, ali
  - $p \wedge q, p \vee q, \neg p$, kjer so p in q formule, ali
  - $\exists X(p(X))$, kjer spr. X prosta v $p(X)$, ali
  - $\forall X(p(X))$, kjer spr. X prosta v $p(X)$
- Uporaba kvantifikatorjev poveže spr. X

QUERY-BY-EXAMPLE (QBE) 28.10.24

Uvod
- „GUI" za poizvedovanje
  - Osnovan na DRR
  - Narejen pred GUI
  - Primeren za enostavna uprašanja

*

DISKI IN DATOTEKE

Diski in datoteke
- SUPB shranjuje podatke na diskih
- To ima veliko posledic na zasnovo SUPB
  - READ: prenos pod. med diskom in RAM
  - WRITE: prenos med RAM in diskom
  - Obe operaciji časovno potratni zato se mora njihova poraba planirati

Zakaj ne vsega shraniti v dinamični spomin
- Prevelika cena
- Dinamični spomin ne ohranja podatkov

*

4.11.24

Pomnilnik SUPB
- Podatki v datotekah
- Znotraj datoteke zapisi, ki so urejeni po straneh
- Dostop preko naslovov
- Vmesni pomnilnik - okvir, kjer so shranjene strani

Ureditev blokov na disku
- Pomemben koncept: „Naslednje" stran
  - Bloki na isti sledi, ki sledijo
  - Bloki na istem cilindru, ki sledijo
  - Bloki na sosednjem cilindru
- Bloki naj bi bili organizirani sekvenčno na disku, da se minimizira čas iskanja in rotacijska zakasnitev
- Sekvenčno skeniranje: branje v naprej, t.j. več sekvenčnih strani se prebere v naprej, pridobitev na času

Delo z diskovnim prostorom
- Nižji nivo SUPB dela z diskovnim pomnilniškim prostorom
- Višji nivoji zahtevajo od nižjega alokacijo/dealokacijo strani in branje in pisanje strani
- Zahteve po sekvenci strani izvršena, da sistem alocira stran v enem sekvenčnem prostoru.

Vmesni pomnilnik v SUPB
- Podatki morajo biti v RAM-u, da SUPB lahko dela z njimi
- Uporablja se tabela parov: <okvir#, stranID>
- Problemi:
  - lahko se hitro napolni

Zahteva po strani
- Če ni v bazeni:
  - Izberi okvir za zamenjavo
  - Če je okvir „umazan" potem se mora zapisati na disk
  - Preberi izbrano stran v izbrani okvir
- Pripni stran in vrni njen naslov

Strategije za zamenjavo strani
- Okvir izbran s strategijo zamenjave
  - LRU, Ura (Clock), MRU, itd.
- Strategija ima lahko velik vpliv na # I/O operacij
- Sekvenčno prelivanje
  - Groda situacija povzročena z LRU + ponavljajoč sekvenčni pregled tabele
  - # okvirjev < # str. datoteke vsaka zahteva povzroči I/O. MRU je v tem primeru boljša

SUPB vs OS file system
- OS ureja diskovni prostor & vmesni pomnilnik
  Zakaj ne bi OS za SUPB urejal ta pravila?
- Razlike v OS podpori: prenosljivost
- Nekatere omejitve, npr. datoteke se ne morajo razteznati preko velikosti diska
- Delo z vmesnim pomnilnikom v SUPB zahteva zmožnost:
  - Pripeti stran v vmesni pomnilnik, prisiliti shranjevanje na disk
  - Prilagoditev strategije zamenjave strani in branje strani v naprej na osnovi obnašanja tipične operacije

Format zapisa: Fiksna dolžina
- Informacija o tipih polja je enaka za vse zapise v datoteki
- Poišči i-to polje ne zahteva pregled vseh zapisov

Format zapisa: Variabilna dolžina
- Dva alternativna formata
- Polja razmejena s posebnim simbolom

*

*

Neurejena datoteka z direktorijem strani
- Zapis pod. strani na glavni strani vsebuje tudi št. prostih zlogov na pod strani
- Direktorij je zbirka strani, implementacija seznamov je ena alternativa

Sistemski katalogi
- Za vsak indeks:
  - Struktura in iskalni ključi
- Za vsako relacijo:
  - Ime, ime datoteke, datotečna struktura
  - Imena in tipi atributov
  - Imena indeksov
  - Integrirane omejitve
- Za vsak pogled
  - Ime pogleda in definicija
- Statistika, avtorizacija, velikost bazena strani

PREGLED INDEKSOV

Podatki na pomnilniških medijih
- Diski imajo direkten dostop do blokov
- Trakovi imajo sekvenčni dostop do strani
- Datotečna organizacija: Organizacija podatkov na pomnilniških medijih
- Arhitektura: Vmesni pomnilnik

Alternativne datotečne organizacije
- Več alternativ
  - Neurejene datoteke
  - Sortirane datoteke
  - Indeksi

Indeksi
- Indeks pohitri iskanje zapisov na osnovi polj iskalnih ključev
- Indeks vsebuje množico podatkovnih vpisov $k^* = k + rid$
- Indeks podpira učinkovito iskanje podatkovnih vpisov k^* z dano vrednostjo ključa k.

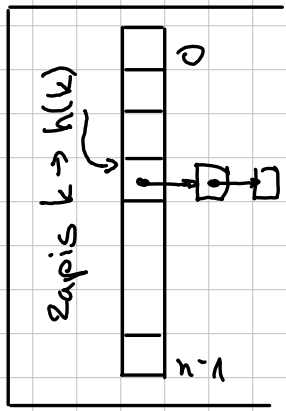
Podatkovni vpis k^*
- Lahko je:
  - Podatkovni zapis z vrednostjo ključa k
  - $<k, rid>$
  - $<k, seznam\ rid>$
- Izbira alt. za pod. vpise je ortogonalna izbiri indeksne tehnike za iskanje vpisov

Podatkovni vpis - možnosti

1. Datotečna struktura, samo en indeks lahko uporabi možnost 1; v primeru da podatkovni zapisi zasedejo veliko prostora, je št. strani, ki vsebujejo podatke, veliko
2.+3. Podatkovni vpisi so tipično precej manjši kot pod. zapis, možnost 3 bolj optimalna od možnosti 2 glede prostora

B+ drevo

- Listi vsebujejo podatkovne vpise, ki so urejeni
- Notranje strani imajo indeksne vpise



Razpršilni indeksi

- Dobra rešitev za selekcijo z enakostjo
- Indeks je zbirka skupkov
  - Skupek = primarna str. + nič ali več prelivnih strani
  - Skupek vsebuje podatkovne vpise
- Razpršilna fja h: $h(r)$ = skupek k, kateremu pripada podatkovni zapis r.
  h uporabi polja iskalnega ključa relacije r

Klasifikacija indeksov

- Primarni vs sekundarni
  - Če iskalni ključ vsebuje primarni ključ potem indeks imenujemo primarni
  - Unique indeks: iskalni ključ je kandidatni ključ
- Povezan vs Nepovezan

*

DREVESNI INDEKSI

Drevesni indeksi
- Drevesno strukturirani indeksi podpirajo iskanje po enakosti in po področju
- ISAM:
  - Statična struktura
  - S časoma drevo postane zelo neuravnoteženo
- B+tree
  - Dinamična struktura
  - Popolnoma uravnoteženo

Iskanje področja
- Če so podatki v urejeni datoteki naredimo binarno iskanje in nato pregled
- cena bin. iskanja je visoka

ISAM
- Indeksna datoteka je vseeno lahko velika
- Listi vsebujejo podatkovne vpise
- Ostajajo prazne prelivne strani

B+tree
- Kompleksnost op. Vstavi/Zbriši = $\log_F n$ (F = faktor, n = št. listov)
- Drevo zelo uravnoteženo
- Indeksne strani vsaj 50% polne

11.11.24

- Vsako vozlišče vsebuje $d \leq m \leq 2d$ vpisov (d imenujemo stopnja drevesa)
- Učinkovito podpira iskanje po enakosti in iskanje področja

Vstavljanje pod. vpisa v B+ drevo
- Poišči list L
- Vstavi pod. vpis v L
  - Če L ni poln končano
  - Če L poln razcepi v L_1 in L_2, prepiši srednji ključ gor in porazdeli ključe
- To se lahko zgodi rekurzivno
- Razcepi širijo drevo, razcep korena zviša drevo

Brisanje pod. vpisa iz B+ drevesa
- Začni pri korenu in poišči list z vpisom
- Izbriši vpis
  - Če L vsaj pol poln zaključi
  - Če vsebuje L d-1 vpisov
    - Poskusimo porazdeliti vse zapise tako, da si jih sposodimo od sosedov
    - Drugače zlij L in soseda
- V primeru zlitja moramo zbrisati zapis, ki kaže na na L ali soseda iz starša

Stiskanje s predponami ključev
- Pomembno je povečati fan-out
- Vrednosti ključev v indeksnih vpisov samo usmerijajo
- Pogosto jih lahko stisnemo
- Vstavljanje/Brisanje mora biti prilagojeno

Masovno polnjenje
- Hitrejše od vstavljanja posameznih el., če je podatkov veliko
- Inicializacija: uredimo podatkovne vpise, dodaj kazalec na prvo stran v novem korenu

RAZPRŠILNI INDEKSI

Uvod
- Razpršilni indeksi so najboljši za iskanje po enakosti
- Ne podpirajo iskanja po področju
- Obstajajo statični in dinamični razpršilni indeksi
- Kompromisi podobni ISAM vs. B+ drevesom

Statični razpršilni indeks
- Primarne strani so fiksne, zasežene sekvenčno ter niso nikoli sproščene
- $h(k) \bmod N$ = skupek, ki kateremu pripada pod. vpis s ključem k (N = # skupkov)
- Skupki vsebujejo pod. vpise
- Razpršilna fja se uporabi na iskalnem ključu. Vrednosti se morajo razpršiti na $[0, N-1]$
  - $h(k) = a \cdot k + b$ → običajno deluje dobro
  - a in b konstanti
- Razvije se lahko dolga veriga preliminh strani, kar poslabša učinkovitost pod. strukture

Razširljivi razpršilni indeks
- Dinamični razpršilni indeks
  - Branje in pisanje skupkov pofratno
- Ideja: Direktorij kazalcev na skupke
  - Podvojimo # skupkov (strani) s podvojitvijo dir in razcepimo samo skupek, ki je napolnjen
  - Dir je veliko manjši kot celotna datoteka zato se podvojitev izvrši hitro (Ni preliminh strani!)
  - Trik je v preureditvi razpršilne fje

Linearni razpršilni indeks
- Shema z dinamičnim razpr. indeksom
- Lin. razp. indeks reši problem z dolgimi prelivnimi stranmi
- Ideja: uporabimo družino razp. fij $h_0, h_1, \ldots$
  - $h_i(key) = h(key) \bmod (2^i N)$; N = začetno št. skupkov
  - h je neka razp. fja (zaloga vrednosti $n_i \in [0, N-1]$)
  - Če $N = 2^{d_0}$, za nek d_0, h_i aplicira h in ogleda zadnjih d_i bitov, kjer $d_i = d_0 + i$ (i = level)
  - h_{i+1} podvoji zalogo vrednosti h_i (podobno podvoj. dir-a)
- Dir-n se izognemo z uporabo prelivnih str. in s cikličnim razcepljanjem skupkov
  - Razcepljanje poteka v rundah (runda se zaključi, ko so vsi skupki razcepljeni)
  - Trenutna št. runde je level
  - Iskanje: Da bi poiskali skupek r, poišči $h_{level}(r)$
    - Če $h_{level}(r)$ je v področju $[Next, N_R]$, smo našli
    - Sicer, r (ali ko pripada skupku $h_{level}(r)$ ali skupku $h_{level}(r) + N_R$. Aplicirati je potrebno $h_{level+1}(r)$, da bi izvedeli

Linearno razpršovanje
- Vstavi: Poišči skupek z uporabo h_{level} / $h_{level+1}$
  - Če skupek poln:
    - dodaj prelivno stran in vstavi
    - (Mogoče) Razcepi Next in ga povečaj (Next += 1)
- Izberemo lahko kateri koli kriterij za razcepljanje
- Ker skupki razcepljeni ciklično se ne morejo razviti dolge prelivne verige
- Podvojevanje dir-a je razširljivem razpršovanju podobno
- Preblop med razpršilnimi fjami je implicitem s povečevanjem

LH je varianta EH
- Shemi podobni
  - Začni z EH indeksom, ki ima dir z N elementi
  - Uporabi prelivne strani in razcepi skupke ciklično
  - Najprej razcepi skupek 0
- Dir se podvojuje postopoma, skupki se hrenirajo fizično zaporedno na disku

EVALUACIJA RELACIJSKIH OPERACIJ

Pogosto uporabljene tehnike
- Algoritmi za evaluacijo:
  - Indeksiranje - za izbor majhnega št. n-teric
  - Heracija - včasih hitreje pregledamo vse n-terice
  - Particije
  - Sortiranje

Statistika in katalogi
- Katalogi vsebujejo vsaj:
  - # zapisov, # strani,

*

Dostopna pot
- Metoda za branje zapisov
  - Pregled datoteke, ali indeksa, ki se ujema s selekcijo

Sortiranje

*

Dva pristopa k splošni selekciji

1. Poišči najbolj selektivno dostopno pot, preberi zapise z izbrano DP in uporabi preostale pogoje, ki se nenjemajo z indeksom
2. Če imamo na razpolago dva ali več indeksov z alternativama (2) ali (3) za pod. vpise