17
hroug

LOGIN
software

software

ORACLE CERTIFIED PARTNER

Alen Prodan

Oracle Balanced Tree Indeksi

software

Agenda

- Uloga indeksa
- Interna struktura B-Tree indeksa
- Block split mehanizam
- Utjecaj obrazaca izmjene podataka na interne strukture B-Tree indeksa

Oracle B-Tree Indeksi

Uloga B-Tree indeksa

- Osnovna namjena je efikasnije pronalaženje redaka u tablicama
- Mogu značajno ubrzati izvođenje SQL naredbi (smanjuje fizički I/O)
- Mogu dramatično i usporiti izvođenje SQL naredbi: povećati korištenje CPU resursa (logički I/O) ili izazvati kontenciju među procesima

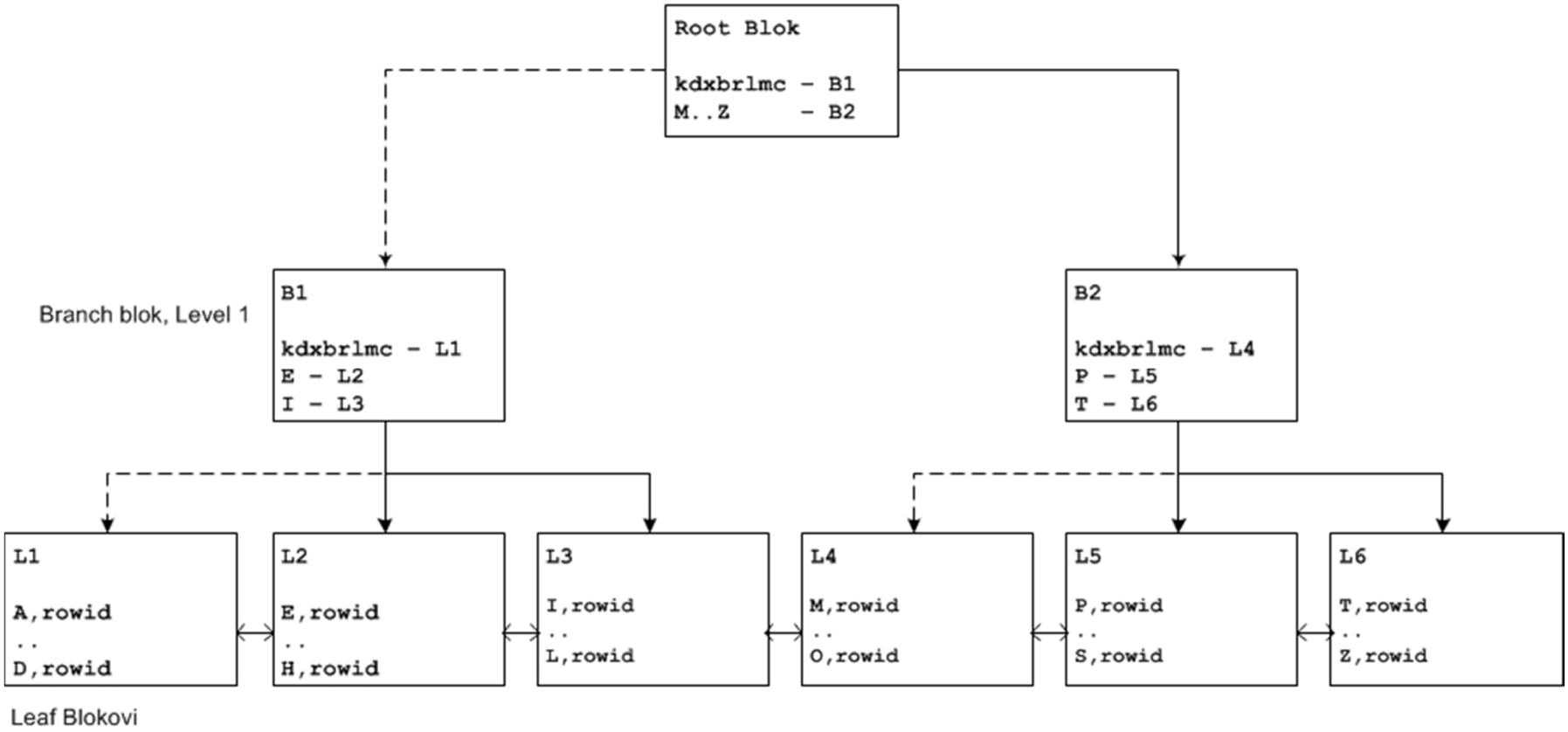
Oracle B-Tree Indeksi

Uvod u B-Tree indekse

- B-Tree indeksi su najčešće korišteni tip indeksa u oracle bazi podataka
- Indeks zapisi su uvijek sortirani prema vrijednostima indeks ključa
- Balanced Tree znači da je indeks uvijek uravnotežen: root block split je jedini trenutak u kojem indeks raste u „visinu” !
- U strukturi B-Tree indeksa pojavljuju se dvije vrste blokova
- Branch blokovi služe za pretraživanje (navigaciju) unutar internih struktura indeksa
- Leaf blokovi sadrže indeks zapise u formi (indeks ključ, rowid)

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa



Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

- Pretraživanje zapisa vrši se navigacijom kroz indeks blokove i to od najviše root blok razine, preko branch do leaf blokova
- Leaf blokovi su međusobno povezani pokazivačima (pointerima) čime tvore double link listu za efikasno pretraživanje putem INDEX RANGE SCAN operacija
- Interna struktura root bloka identična je strukturi branch bloka, osim kada se indeks sastoji isključivo od jednog bloka – tada je interno formatiran kao leaf blok
- Visina (height) indeksa označava broj blokova koje je potrebno prijeći od root do leaf bloka
- Branch Level (blevel) izražava se kao visina umanjena za 1 (razinu leaf blokova)

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

- Branch blokovi sadrže parcijalne indeks zapise – minimalna informacija potrebna za usmjeravanje pretrage kroz strukture indeksa
- Branch zapisi su u formatu (parcijalni indeks ključ, adresa bloka)
- PCTFREE nema utjecaja na branch blokove, već samo na leaf blokove
- Dobiva sa na većoj gustoći zapisa unutar branch blokova
- Parcijalni indeks zapisi pohranjeni su u row directory branch bloka

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

- Detaljniji uvid u strukturu indeks stabla dobivamo koristeći treedump dijagnostički event:

```
select object_id, object_type from user_objects where object_name = 'T5_DATUM' ;

 OBJECT_ID OBJECT_TYPE
---------- -------------------
     73444 INDEX
alter session set events 'immediate trace name treedump level 73444';
```

Usporedbe radi priložen je rezultat analyze index naredbe:

```
select br_blks, lf_blks, br_rows, lf_rows, pct_used from index_stats;

   BR_BLKS    LF_BLKS    BR_ROWS    LF_ROWS   PCT_USED
---------- ---------- ---------- ---------- ----------
         1         10    !=    9       3650         80  <= 9 BR_ROWS, a ima 10 LF_BLKS
```

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

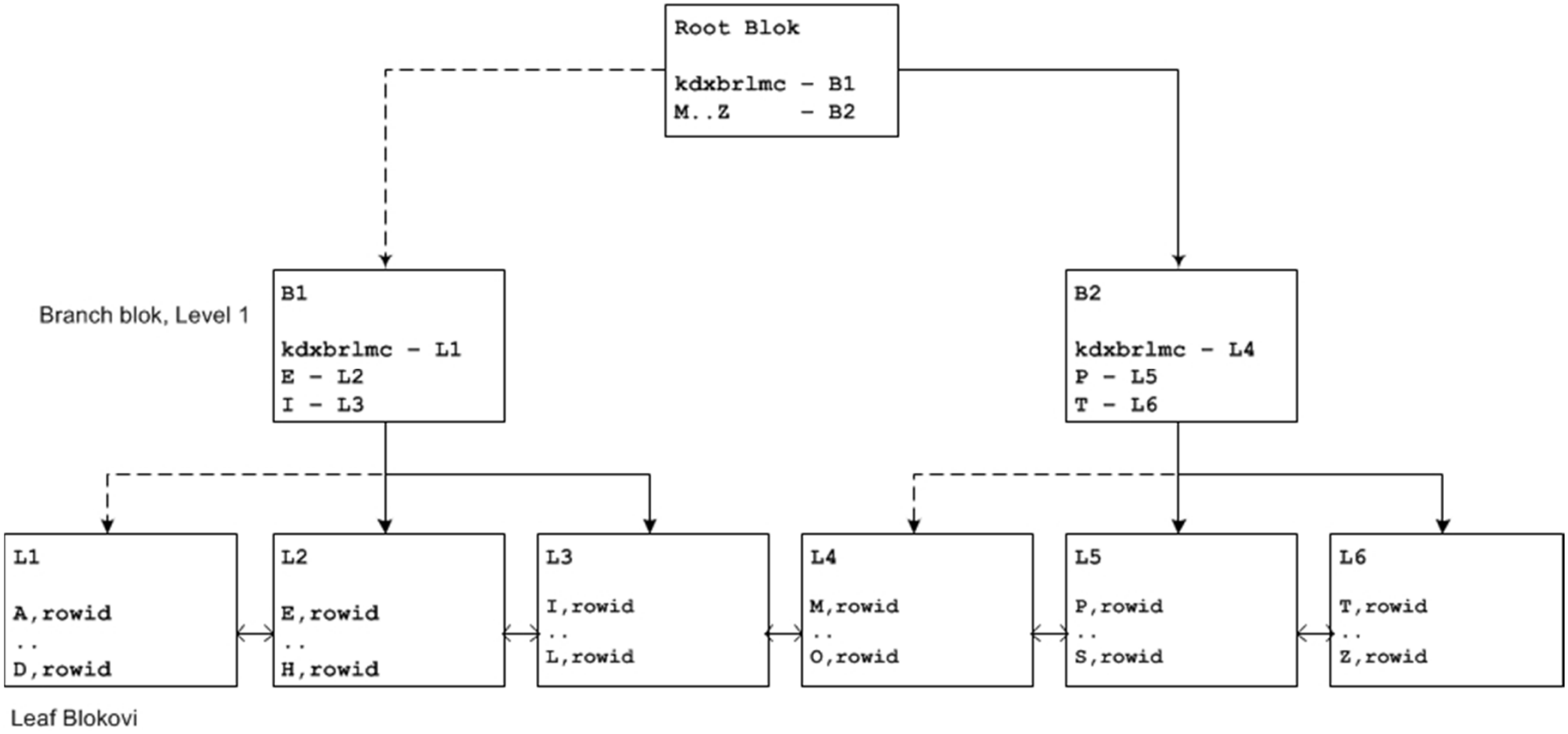
- Rezultat treedump naredbe pronalazimo u serverskoj trace datoteci:

```
 ----- begin tree dump
branch: 0x1004fe1 16797665 (0: nrow: 10, level: 1)
   leaf: 0x1004fe2 16797666 (-1: nrow: 377 rrow: 377) /* kdxbrlmc = (-1) leftmost child
   leaf: 0x1004fe3 16797667 (0: nrow: 377 rrow: 377)  /* rrow= tekući broj redaka */
   leaf: 0x1004fe4 16797668 (1: nrow: 377 rrow: 377)  /* nrow= broj redaka u row dir */
   leaf: 0x1004fe5 16797669 (2: nrow: 377 rrow: 377)
   leaf: 0x1004fe6 16797670 (3: nrow: 377 rrow: 377)
   leaf: 0x1004fe7 16797671 (4: nrow: 377 rrow: 377)
   leaf: 0x1004fe8 16797672 (5: nrow: 377 rrow: 377)
   leaf: 0x1004fe9 16797673 (6: nrow: 377 rrow: 377)
   leaf: 0x1004fea 16797674 (7: nrow: 377 rrow: 377)
   leaf: 0x1004feb 16797675 (8: nrow: 257 rrow: 257)
----- end tree dump
```

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

- Leftmost child nije zapisan u row directory i ne postoji branch zapis (parcijalni ključ, adresa), već se izvede lokacija pomoću parent bloka:

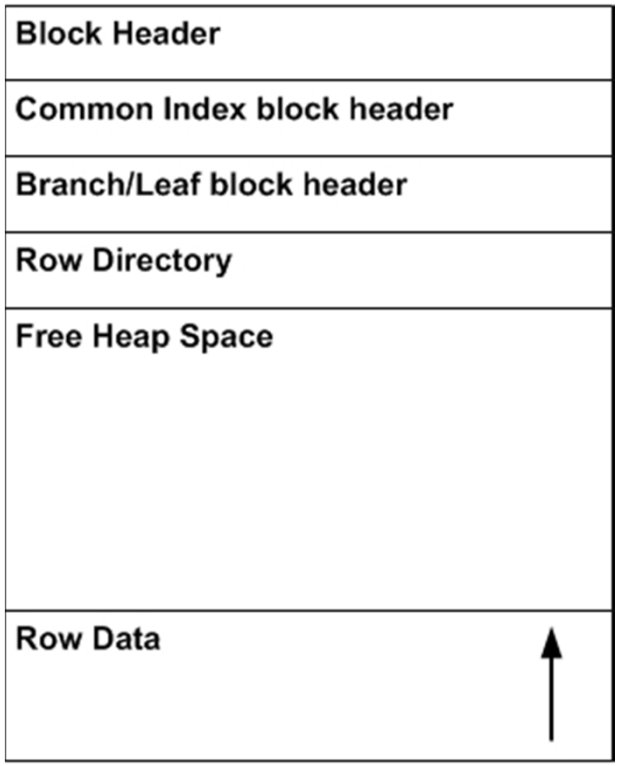


Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

- Više detalja o internoj strukturi branch/leaf blokova dobivamo putem simboličkog dumpa

```
 ----- begin tree dump
branch: 0x1004fe1 16797665 (0: nrow: 10, level: 1)
...
----- end tree dump

select
dbms_utility.data_block_address_file(16797665)||'/'||
dbms_utility.data_block_address_block(16797665)
fno_blk from dual;

FNO_BLK
--------------------------------------------------------------------------------
4/20449

alter system dump datafile 4 block 20449;
```



Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

Simbolički prikaz indeks branch bloka – Transaction control:

```
Block header dump:  0x01004fe1
 Object id on Block? Y
 seg/obj: 0x11ee4  csc: 0x00.1dcd29  itc: 1  flg: -  typ: 2 - INDEX
     fsl: 0  fnx: 0x0 ver: 0x01

  Itl           Xid                  Uba         Flag  Lck        Scn/Fsc
0x01   0xffff.000.00000000  0x00000000.0000.00  C---    0  scn 0x0000.001dcd29
```

seg/obj – object id
csc – commit scn
itc – broj elemenata u ITL tablici (default: branch 1, leaf 2 ITL slota)
typ – oznaka vrste bloka (2 – INDEX blok)
Itl – pozicija, index u ITL arrayu
Xid – id transackcije u formatu undo segment.undo slot.undo sequence number
Uba – blok adresa.block sequence number.undo record unutar bloka
Flag – status trenutne transakcije (C---) = commited & cleaned out
Lck – broj redaka koje je transakcija izmjenila (zaključala) u ovom bloku
Scn/Fsc – System Change Number kada je transakcija potvrđena (commit)

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

Simbolički prikaz indeks branch bloka - Branch header sekcija:

```
Branch block dump
=================
header address 47010060950084=0x2ac161fbda44
kdxcolev 1   /* oznaka razine bloka: first level branch blok (1), leaf blok (0) */
KDXCOLEV Flags = - - -
kdxcolok 0   /* označuje da li je u tijeku strukturalna transakcija nad blokom */
kdxcoopc 0x80: opcode=0: iot flags=--- is converted=Y /* interna oznaka operacije */
kdxconco 2 /* broj stupaca u indeksu */
kdxcosdc 0 /* broj strukturalnih promjena u indeksu koje su obuhvatile ovaj blok */
kdxconro 9 /* broj indeks zapisa (ne uključuje kdxbrlmc leftmost child pointer) */
kdxcofbo 46=0x2e /* offset početka slobodnog heap prostora */
kdxcofeo 7885=0x1ecd /* offset završetka slobodnog heap prostora */
kdxcoavs 7839 /* raspoloživi prostor za pohranu unutar bloka (7885 – 46 = 7839) */
kdxbrlmc 16797666=0x1004fe2 /* pokazivač na leftmost child blok (-1) */
kdxbrsno 0
kdxbrbksz 8056 /* ukupna veličina korisnog prostora u bloku, overhead prethodno oduzet
kdxbr2urrc 0
```

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

Simbolički prikaz indeks branch bloka – Row Data sekcija:

```
...
row#0[8037] dba: 16797667=0x1004fe3 /* pokazivač na drugi leaf blok i indeks ključ */
col 0; len 7; (7):  78 70 01 02 01 01 01
col 1; len 6; (6):  01 00 4f d1 01 79
row#1[8018] dba: 16797668=0x1004fe4 /* pokazivač na treći leaf blok i indeks ključ */
col 0; len 7; (7):  78 70 01 03 01 01 01
col 1; len 6; (6):  01 00 4f d2 01 41
....

 ----- begin tree dump
branch: 0x1004fe1 16797665 (0: nrow: 10, level: 1)
   leaf: 0x1004fe2 16797666 (-1: nrow: 377 rrow: 377)
   leaf: 0x1004fe3 16797667 (0: nrow: 377 rrow: 377)
   leaf: 0x1004fe4 16797668 (1: nrow: 377 rrow: 377)
...
```

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

Simbolički prikaz indeks Leaf bloka – Transaction control:

```
Block header dump:  0x01004fe2
 Object id on Block? Y
 seg/obj: 0x11eea  csc: 0x00.1e099e  itc: 2  flg: -  typ: 2 - INDEX
     fsl: 0  fnx: 0x0 ver: 0x01

 Itl           Xid                  Uba         Flag  Lck        Scn/Fsc
0x01   0x0000.000.00000000  0x00000000.0000.00  ----    0  fsc 0x0000.00000000
0x02   0xffff.000.00000000  0x00000000.0000.00  C---    0  scn 0x0000.001e099e
```

- Leaf indeks blokovi imaju dodijeljena 2 ITL slota (default)
- ITL element 0x01 rezerviran je za blok split transakcije

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

Simbolički prikaz indeks Leaf bloka – Leaf index block header

Pojavljuju se zapisi koji nisu postojali u branch blok header sekciji:

```
Leaf block dump
===============
...
kdxlende 0 /* broj izbrisanih indeks zapisa */
kdxlenxt 16797667=0x1004fe3 /* pointer prema sljedećem leaf bloku */
kdxleprv 0=0x0 /* pointer prema prethodnom leaf bloku */
kdxlebksz 8032 /* ukupna količina korisnog prostora u bloku (za 24 bytea manja nego kod branch bloka zbog postojanja 2 ITL slota) */
```

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

Simbolički prikaz indeks Leaf bloka – Row Data sekcija

Struktura indeks zapisa unutar leaf bloka za non-unique indeks:

```
row#0[8015] flag: ------, lock: 0, len=17
col 0; len 7; (7):  78 70 01 01 01 01 01 /* datumsko polje, 7-byteova */
col 1; len 6; (6):  01 00 4f d1 00 00    /* rowid, automatski postaje sastavni dio
                                            index ključa i garantira uniqueness */
```

Struktura indeks zapisa unutar leaf-bloka za unique indeks:

```
row#0[8016] flag: ------, lock: 0, len=16, data:(6):  01 00 4f d1 00 00  <= rowid
col 0; len 7; (7):  78 70 01 02 01 01 01 /* indeksirani stupac tablice, 7-byte date */
```

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

- Zapisi u indeks blokovima pohranjuju se pri dnu slobodnog heap prostora unutar bloka indeksa te se evidentiraju u row directory
- Za razliku od tablica, elementi unutar row directoryija sortiranu su redoslijedno prema vrijednosti indeks ključa
- To omogućuje pretraživanje indeks ključeva putem algoritma za binarno pretraživanje (binary search)
- Binary search u svakoj iteraciji dijeli na pola skup elemenata kojega pretražuje, pa vrijeme odziva pretraživanja raste logaritamskom skalom koja se izražava kao O (log (2,N))
- Kod eksponencijalnog rasta količine podataka, vrijeme odziva raste linearno

Oracle B-Tree Indeksi

Interne strukture B-Tree indeksa

- Na primjer: za pronalaženje traženog zapisa unutar leaf blok sa 377 indeks redaka potrebno je posjetiti maksimalno 9 elemenata budući je log(2, 377) = 9
- Sekvencijalno pretraživanje kojega označujemo kao O (N) zahtijevalo bi posjetu svakom od 377 elemenata
- Stoga, binarno pretraživanje zahtijeva tek 9 / 377 * 100 = 2,38% resursa u odnosu na sekvencijalno pretraživanje

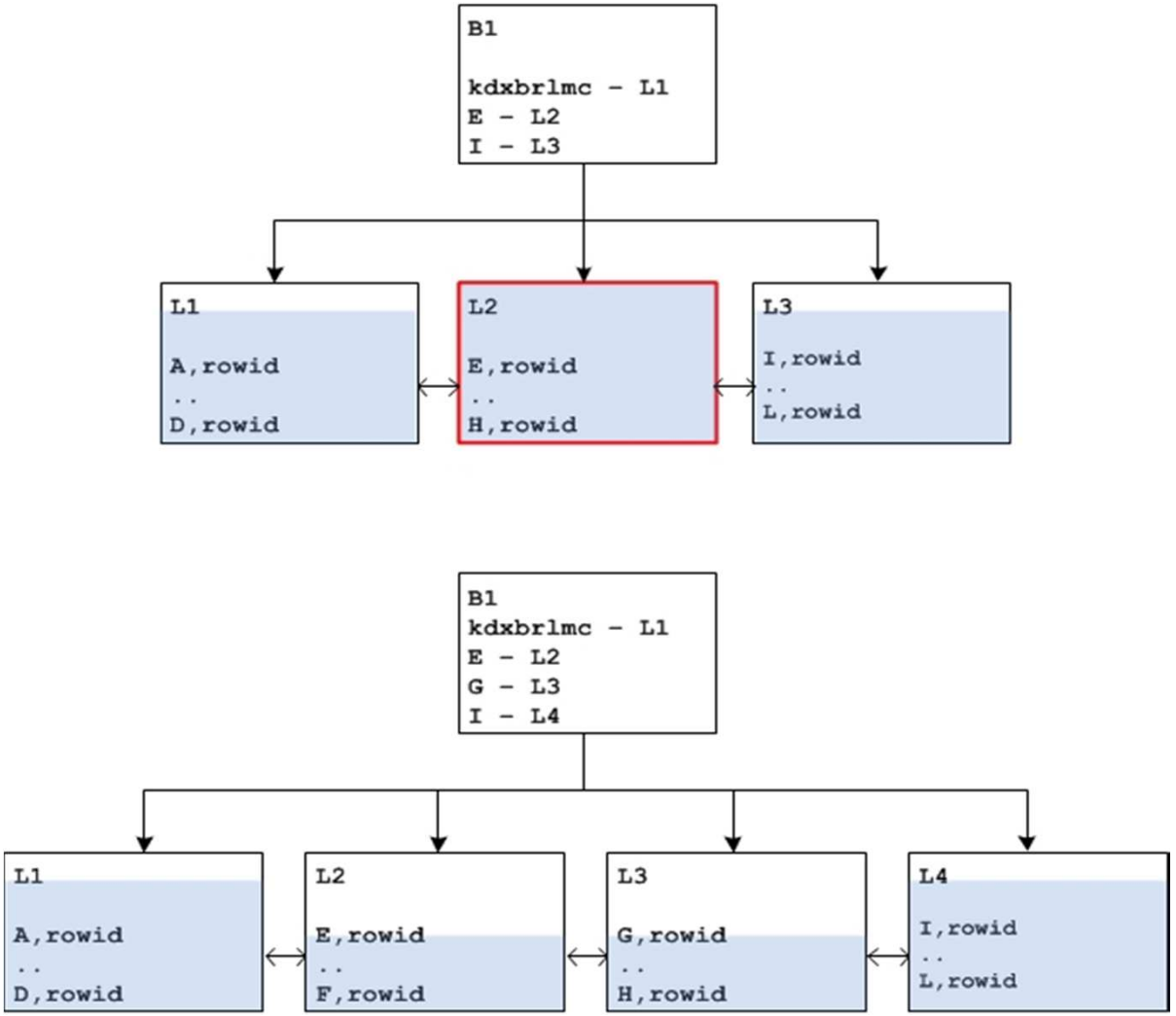
Oracle B-Tree Indeksi

Blok split mehanizam

- Prilikom kreiranja ili rekreiranja indeksa, u svakom leaf bloku rezervirano je PCTFREE (default 10%) prostora za naknadni upis redaka u taj indeks blok
- Kada se iscrpi sav slobodni prostor unutar bloka, događa se dioba bloka – blok split
- Dva su osnovna modaliteta diobe blokova: 50-50 blok split i 90-10 blok split

Oracle B-Tree Indeksi

Blok split mehanizam – 50-50 blok split



Oracle B-Tree Indeksi

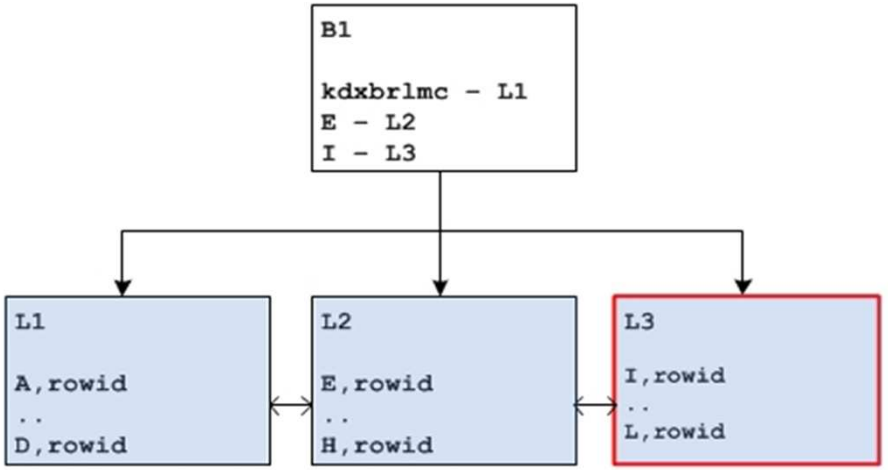
Blok split mehanizam – 50-50 blok split

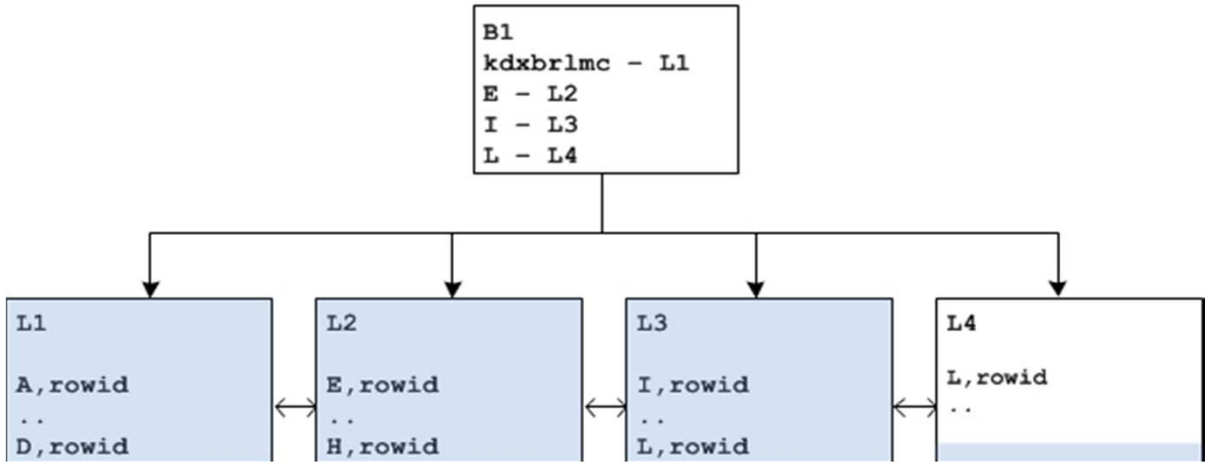
- Konceptualno 50-50 blok split izvodi se na sljedeći način:

1. Indeksu se dodjeljuje novi blok, preuzet sa freeliste
2. Preraspodijeljuju se podaci u bloku koji se dijeli na način da doljnja polovica (po volumenu) ostaje u postojećem bloku, a gornja polovica kopira se u novi blok
3. Novi indeks redak pohranjuje se u odgovarajući blok
4. kdxlenxt pokazivač originalnog bloka pokazuje na novi blok
5. kdxleprv desnog susjednog bloka pokazuje na novi blok
6. Ažurira se branch blok te se dodaje novi zapis koji referencira min(indeks ključ) vrijednost u novom leaf bloku
7. Ako u branch bloku iz koraka 6. nema mjesta, vrši se 50-50 branch blok split

Oracle B-Tree Indeksi

Blok split mehanizam – 90-10 blok split





Oracle B-Tree Indeksi

Blok split mehanizam – 50/50 blok split

- Konceptualno 90-10 blok split izvodi se na sljedeći način:

1. Ukoliko je novi indeks redak jednak ili veći od trenutne maksimalne vrijednosti indeks ključa, tada se izvodi 90-10 block split operacija
2. 90-10 indeks blok split je tipičan scenario koji spriječava fragmentaciju prostora u leaf blokovima u koje se pohranjuju zapisi sa rastućim nizom vrijednosti (npr. oracle sequence)
3. Naziv 90-10 blok split nije sasvim korektan, budući da se u novi blok prenosi manje od 10% podataka (bliže je iznosu od 1%)

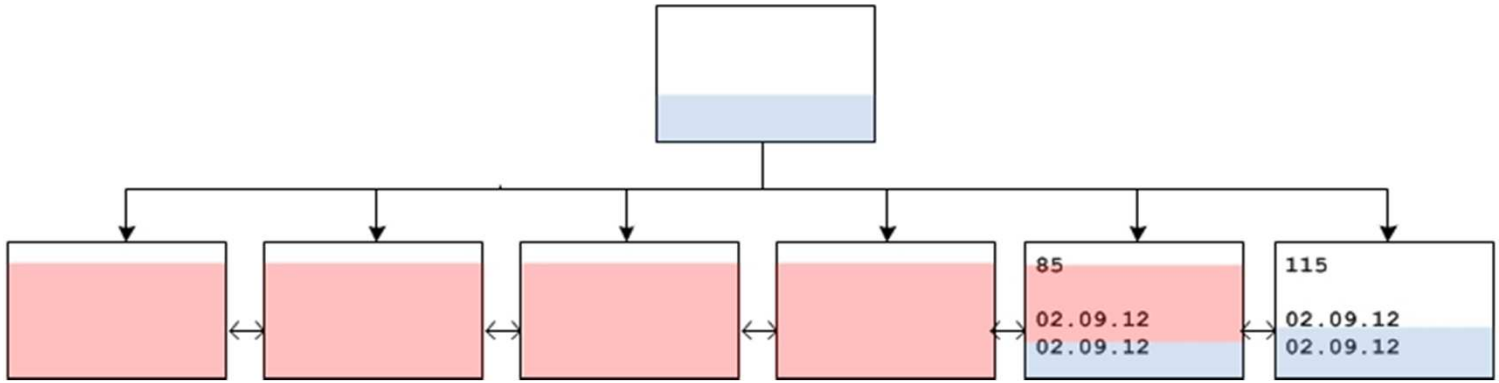
Sustav će primjeniti 90-10 blok split diobu samo ukoliko je vrijednost koja se upisuje jednaka ili veća od trenutne maksimalne vrijednosti. U svim ostalim slučajevima koristi se 50-50 blok split mehanizam.

Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 1.: brisanje indeks blokova u lijevom dijelu indeksa, te insert podataka u desnom dijelu indeksa


```
delete from t where datum < to_date('02.09.2012', 'dd.mm.yyyy');
```



Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

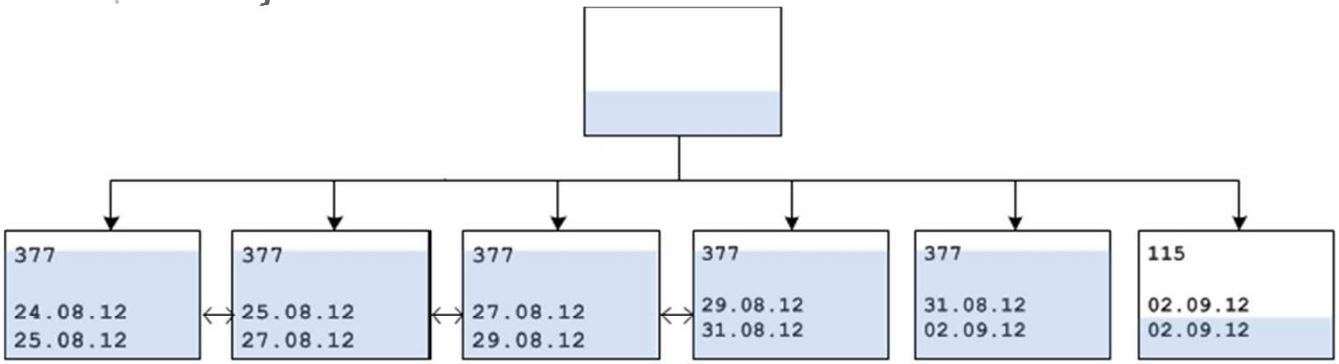
- PRIMJER 1.: brisanje indeks blokova u lijevom dijelu indeksa, te insert podataka u desnom dijelu indeksa
- Rješenje: rebuild, coalesce, shrink

```
select min(datum) from t;

MIN(DATU
--------
02.09.12

 ------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Rows  | Bytes | Cost (%CPU)|
------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         |     1 |     8 |     2   (0)|
|   1 |  SORT AGGREGATE              |         |     1 |     8 |            |
|   2 |   INDEX FULL SCAN (MIN/MAX)| T_DATUM |     1 |     8 |     2   (0)|
------------------------------------------------------------------------------

 Statistics
----------------------------------------------------------

            ...
          6  consistent gets   <= SQL query je posjetio svih 6 leaf blokova
```

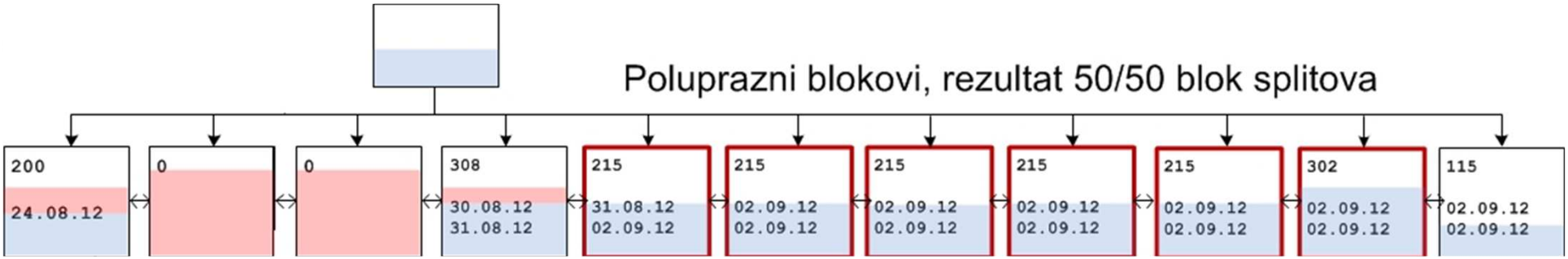
Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 2.: fragmentacija zbog ažuriranja većeg broja identičnih vrijednosti




```
update t
set datum = to_date('02.09.2012', 'dd.mm.yyyy')
where datum between to_date('25.08.2012', 'dd.mm.yyyy')
            and to_date('29.08.2012', 'dd.mm.yyyy');
```



Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 3.: utjecaj redoslijeda upisa podataka na fragmentaciju prostora unutar leaf blokova

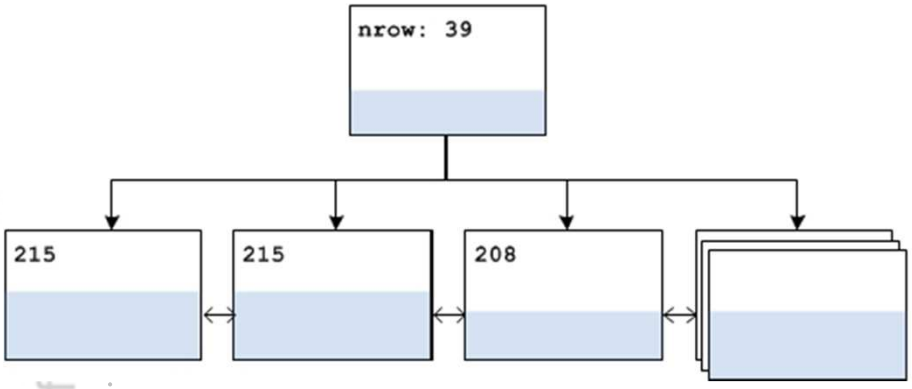
```
create table t
(
 id number,
 d1 date,
 d2 date
);

create index t_d1 on t(d1);
create index t_d2 on t(d2);

insert into t
select rownum,
       to_date('01.01.2012', 'dd.mm.yyyy')+mod(rownum, 10) d1,
       to_date('01.01.2012', 'dd.mm.yyyy')+trunc((rownum-1)/100) d2
from dual
connect by level <=10000; commit;
```

Oracle B-Tree Indeksi

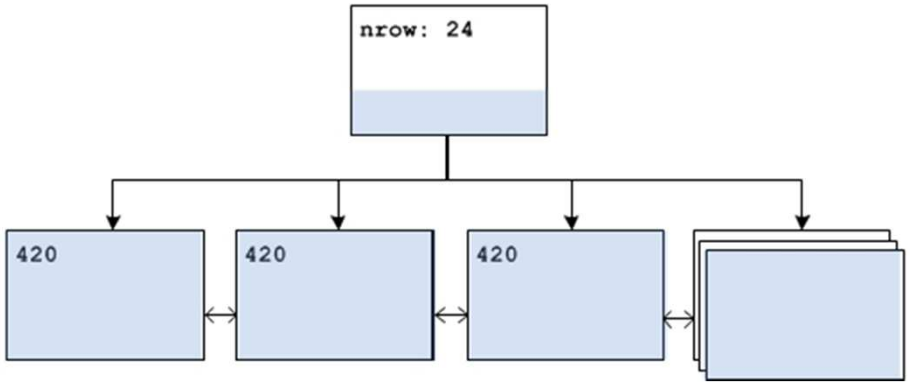
Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 3.: utjecaj redoslijeda upisa podataka na fragmentaciju prostora unutar leaf blokova

```
INDEKS: T_D1, cikličke izmjene dolaznih vrijednosti

----- begin tree dump T_D1
branch: 0x1005141 16798017 (0: nrow: 39, level: 1)
   leaf: 0x1005142 16798018 (-1: nrow: 215 rrow: 215)
   leaf: 0x1005168 16798056 (0: nrow: 215 rrow: 215)
   leaf: 0x1005179 16798073 (1: nrow: 208 rrow: 208)
   ....
```




```
INDEKS: T_D2, monotoni rast dolaznih vrijednosti

----- begin tree dump T_D2
branch: 0x1005149 16798025 (0: nrow: 24, level: 1)
   leaf: 0x100514a 16798026 (-1: nrow: 420 rrow: 420)
   leaf: 0x100514b 16798027 (0: nrow: 420 rrow: 420)
   leaf: 0x100514c 16798028 (1: nrow: 420 rrow: 420)
....
```



Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 4.: utjecaj PCTFREE atributa na fragmentaciju prostora unutar leaf blokova

```
create table t
(
 n1 number,
 d1 date,
 d2 date
);

insert into t
select rownum, trunc(sysdate, 'yyyy')+rownum d1, trunc(sysdate, 'yyyy')+rownum d2
from dual
connect by level <= 3650;

commit;
```

Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 4.: utjecaj PCTFREE atributa na fragmentaciju prostora unutar leaf blokova

```
create index t_d1 on t(d1);
create index t_d2 on t(d2) pctfree 50;

analyze index t_d1 validate structure;
select height, br_blks, lf_blks, br_rows, lf_rows from index_stats;

    HEIGHT    BR_BLKS    LF_BLKS    BR_ROWS    LF_ROWS
---------- ---------- ---------- ---------- ----------
         2          1         10          9       3650

analyze index t_d2 validate structure;
select height, br_blks, lf_blks, br_rows, lf_rows from index_stats;

    HEIGHT    BR_BLKS    LF_BLKS    BR_ROWS    LF_ROWS
---------- ---------- ---------- ---------- ----------
         2          1         18         17       3650
```

Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 5.: utjecaj broja istovremenih transakcija nad istim indeks blokom na raspoloživost slobodnog prostora u leaf blokovima
- Pitanje: što bi se dogodilo kada bi 200 sesija istovremeno imale otvorenu transakciju nad istim indeks blokom ?
- Test je izvršen na način da je putem database schedulera (DBMS_JOB paket) pokrenuto 200 sesija
- Svaka sesija insertirala je jedan redak u tablicu, ali nije odmah izvršila commit;
- Za vrijeme testa 32 sesije ostale su blokirane na wait eventu:

```
EVENT                                                          COUNT(*)
-------------------------------------------------------------- ----------
enq: TX - allocate ITL entry                                           32
```

- Po završetku masivnog inserta, indeks je i dalje sadržavao samo jedan leaf blok:

```
----- begin tree dump
leaf: 0x1005141 16798017 (0: nrow: 200 rrow: 200)
----- end tree dump
```

Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 5.: utjecaj broja istovremenih transakcija nad istim indeks blokom na raspoloživost slobodnog prostora u leaf blokovima
- Simbolički dump otkriva da su sesije za vrijeme testa alocirale 167 novih ITL slotova, ukupno je dosegnut maksimum od 169 ITL (hard limit za 8 KB veličinu bloka):

```
Block header dump:  0x01005141
 Object id on Block? Y
 seg/obj: 0x11f9b  csc: 0x00.1fa843  itc: 169  flg: -  typ: 2 - INDEX
     fsl: 0  fnx: 0x0 ver: 0x01

 Itl           Xid                  Uba         Flag  Lck        Scn/Fsc
0x01   0x0000.000.00000000  0x00000000.0000.00  ----    0  fsc 0x0000.00000000
0x02   0x00d3.001.00000002  0x01403a52.0000.03  C---    0  scn 0x0000.001fa687
0x03   0x00d9.001.00000002  0x01403ab2.0000.03  C---    0  scn 0x0000.001fa6c2
0x04   0x0026.002.000000f0  0x01400d2e.0041.23  C---    0  scn 0x0000.001fa572
0x05   0x0023.000.000000f2  0x01400368.0058.36  C---    0  scn 0x0000.001fa578
...
0xa9   0x00c8.001.00000002  0x014018a2.0000.03  C---    0  scn 0x0000.001fa559
```

Oracle B-Tree Indeksi

Utjecaj izmjene podataka na interne B-Tree strukture

- PRIMJER 5.: utjecaj broja istovremenih transakcija nad istim indeks blokom na raspoloživost slobodnog prostora u leaf blokovima
- PROBLEM: nakon blok splita nad ovim indeks blokom, ITL slotovi se kopiraju u idući blok, te se otvara mogućnost propagiranja ekspandiranog broja ITL slotova i kroz ostatak strukture indeksa
- To bi dramatično smanjilo ukupni iskoristivi prostor u indeks blokovima, budući da svaki ITL slot zauzima 24 bytea, pa je 169 x 24 = 4.056 byte!
- Ukoliko je ekspanzija ITL slotova izazvana specijalnim slučajem, te ostali blokovi neće biti podložni takvoj razini konkurentnosti izmjene podataka, rebuild indeksa može spriječiti nepotreban gubitak iskoristivog prostora za pohranu novih zapisa

Oracle B-Tree Indeksi

Rebuild/coalesce indeksa – da ili ne ?

- Ne postoji jednostavno pravilo na temelju kojega odlučiti
- Najbolje temeljiti odluku na utjecaju kojega fragmentacija indeks blokova ima na SQL operacije
- Rebuild indeksa smanjuje broj leaf_blokova i rjeđe blevel, ali nema utjecaja na clustering_factor, pa je onda i manji utjecaj na povećeanje sklonosti optimizera prema korištenju indeksa
- Da bi utjecaj bio vidljiv treba cardinality SQL upita biti velik, ali onda u pravilu optimizer ima sklonost prema FTS operacijama

```
Cost =
 blevel +
 ceil(leaf_blocks * index selectivity)+
 ceil(clustering_factor * table_selectivity)
```

Pitanja i Odgovori

software