Prednáška 9

Indexy, Transakcie

Obsah prednášky

- Organizácia
- Indexy
  - Pokračovanie z minulej prednášky
- Súbežné spracovanie
  - Transakcie

Organizácia

- Zadanie 5

Indexy

Pripomenutie

- B+tree index
  - Ukážka princípu a tiež jeho analýza vzhľadom na stránky

Indexy - typy

- **Hash index**
- **B+tree**
- **Bitmap index**
- Ostané typy indexov – **inverted**, **trie**, **radix** ...

Pokročilé databázové technológie
Inžiniersky stupeň

- PostgreSQL
  - **B-tree**, **hash**, **GiST**, **SP-GiST**, **GIN** a **BRIN** (dokumentácia - verzia 13 - release date 24 September 2020)

Priniesla optimalizáciu veľkosti indexov

B+tree: v praxi

- Ak **rad–d** B+tree je 100
  - Počet potomkov **m** pre daný uzol je $d \leq m \leq 2d$
- Typický fill factor je 67%
  - Potom priemerný počet potomkov sa bude pohybovať **133**
- Kapacita B+tree
  - 3 úroveň: $133^3 = 2\ 352\ 637$ záznamov
  - 4 úroveň: $133^4 = 312\ 900\ 700$ záznamov
- Možnosť uloženia stránok v rámci buffer poola
  - 1 level 1 stránka = 8KB
  - 2 level 133 stránok = 1MB
  - 3 level 17689 stránok = 133MB

B+tree: duplicita kľúčov

- Viacero záznamov obsahuje rovnaký kľúč
- Možnosti ako riešiť
  - Záznamy s rovnakým kľúčom sú uložené v rámci jednej stránky + overflow page
  - Záznamy s rovnakým kľúčom sú uložené v rámci listov
    - Nutnosť modifikovať prehľadávanie v rámci listov

B+tree vs B-tree

Difference Between B-Tree And B+ Tree

B-Tree	B+ Tree
Data is stored in leaf nodes as well as internal nodes.	Data is stored only in leaf nodes.
Searching is a bit slower as data is stored in internal as well as leaf nodes.	Searching is faster as the data is stored only in the leaf nodes.
No redundant search keys are present.	Redundant search keys may be present.
Deletion operation is complex.	Deletion operation is easy as data can be directly deleted from the leaf nodes.
Leaf nodes cannot be linked together.	Leaf nodes are linked together to form a linked list.

Zdroj : https://www.softwaretestinghelp.com/b-tree-data-structure-cpp/

B+tree: leaf nodes - hodnoty

- Hodnoty v rámci leaf node
- Prístup 1. – Record ID
  - Ukazovateľ na lokalitu záznamu tabuľky, pre ktorý korešponduje záznam v indexe
- Prístup 2. – Dáta záznamu
  - Obsahuje samotný záznamu tabuľky
    - InnoDB v rámci MySQL vytvára b+tree na základe primárneho kľúča
  - V prípade použitia dalších indexov sa dané indexy odkazujú na record ID

Hash index

- Využívanie hash funkcie pre výpočet pozície, kde bude záznam uložený
  - Dôležitosť výber vhodnej funkcie, ktorá je rýchla a ma primeraný počet kolízii pre dané dáta
- Efektívny pre rovnosť (equality search)
- Nemožno použiť pri range prehľadávaní (range search) alebo partial scan
- Časová zložitosť O(1) + prehľadávanie prípadných overflow pages

- Typy Hash indexov
  - Statické
  - Dynamické (extendible, dynamic)

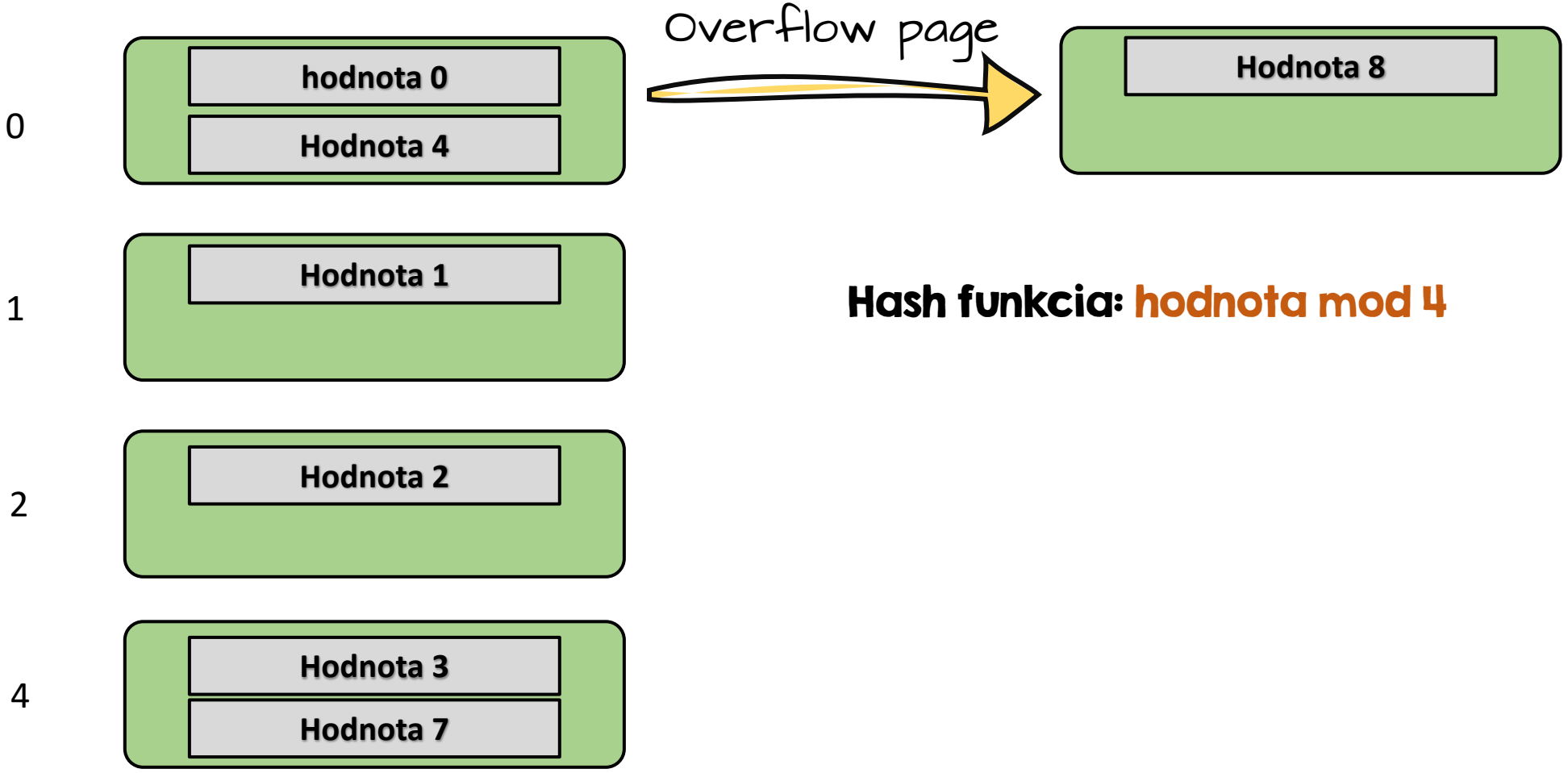
Static Hash Index

- Je kolekcia bucket
  - Bucket – primárna strának + overflow stránka
  - Každý bucket – obsahuje viacero záznamov

- **h**(k) mod **N**
  - N - počet bucketov

- Záznamy s rôznymi kľúčmi môžu byť umiestnené v rámci to istého bucketu
  - V prípade zaplnenia primárnej stránky + potreba overflow page

Hash index - statické



Hash funkcia: hodnota mod 4

Hash index vs b+tree index

- Hash
  - Vhodný na hľadanie s rovnosťou (=)
  - Konštatnej časovej zložitosti pre vkladanie a hľadanie

- B+tree
  - Vhodný pre hľadanie rovnosti ale aj rozsahu

Bitmap index

- Špeciálny typ indexu
- Každý bitmap index je vytvorený nad jedným hľadaným kľučom
  - Napr. žena, muž
- Vhodný pre low cardinality
- Drahé vkladanie a mazanie záznamov
  - Nutnosť úpravy bitmapy

- Využíva sa vo forme in-memory indexu, ktorý je dočasne vytvorený len pre daný dopyt (query)
  - Napr. v rámci Bitmap scan

Bitmap index

Record number	ID	Gender	Income_level
0	11111	m	L1
1	22222	f	L2
2	12345	f	L1
3	89879	m	L4
4	12345	f	L3

Bitmap index pre atribút Gender

m	**10010**
w	**01101**

Bitmap index pre atribút Income_level

L1	**10100**
L2	**01000**
L3	**00001**
L4	**00010**
L5	**00000**

Ako vyzerá vyhľadávanie

- Index access – prechod stromom k jednotlivým listom
  - toto nie je problém, keďže strom je balanced

- Index range scan – prehľadávanie zoznamu listov (môže prejsť veľkú časť indexu)
  - ak musí prejsť veľa tak nastáva problém spojený s prístupom k tabuľkám (Table access), čo spôsobí spomalenie

- Tabble access – vytiahnutie dát z tabuľky
  - Nastáva problém pokiaľ je treba prejsť veľa tabuliek

PostgreSQL - prístupové metódy

- Seq Scan
- Index scan
- Bitmap scan
- Hash scan

Indexy

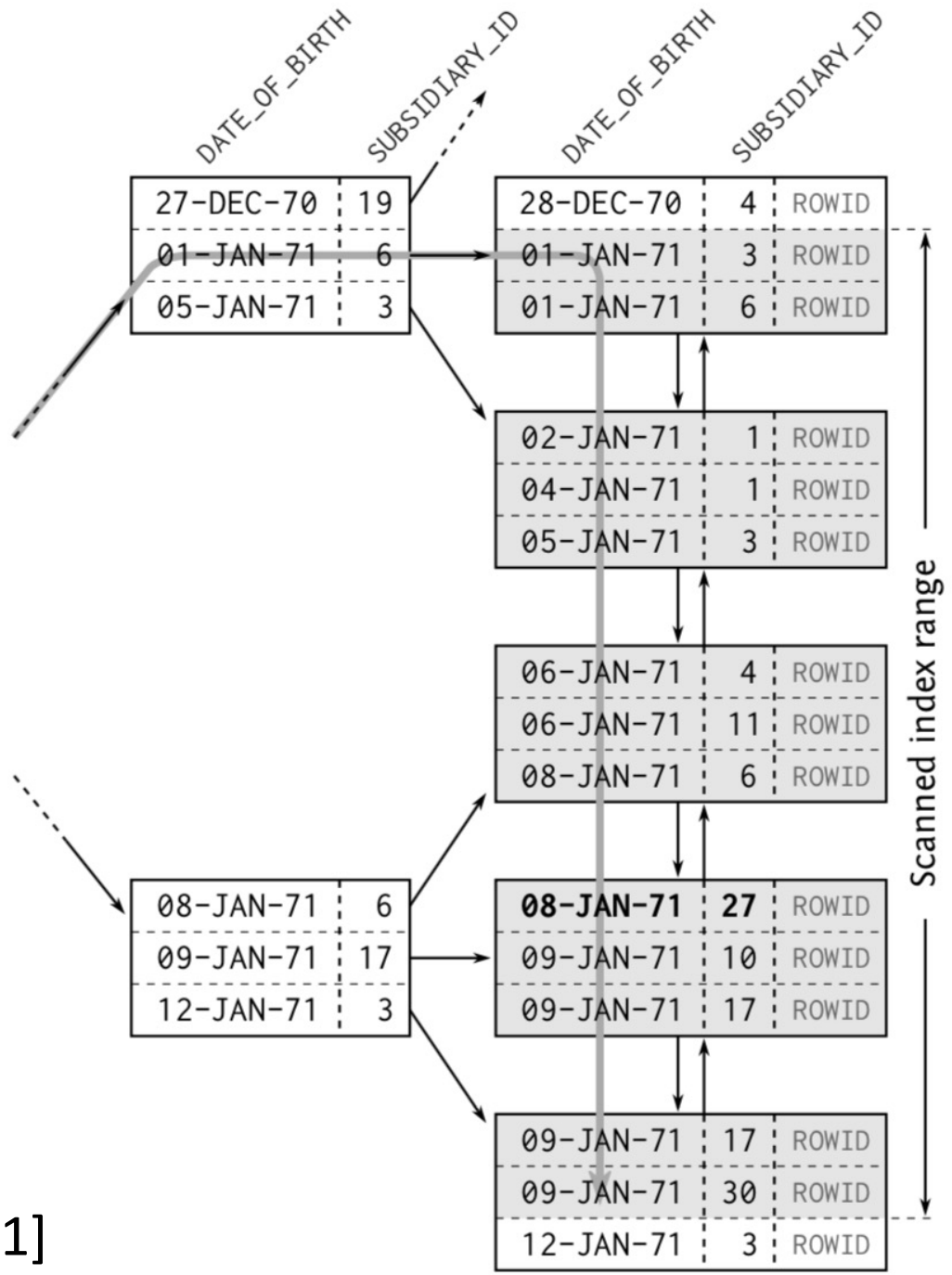
- Vytvorenie indexu neprináša iba pozitívne vlastnosti
  - Spomalenie pri manipulácií dát
  - Zvýšenie požiadaviek na úložisko
  - Údržba indexov

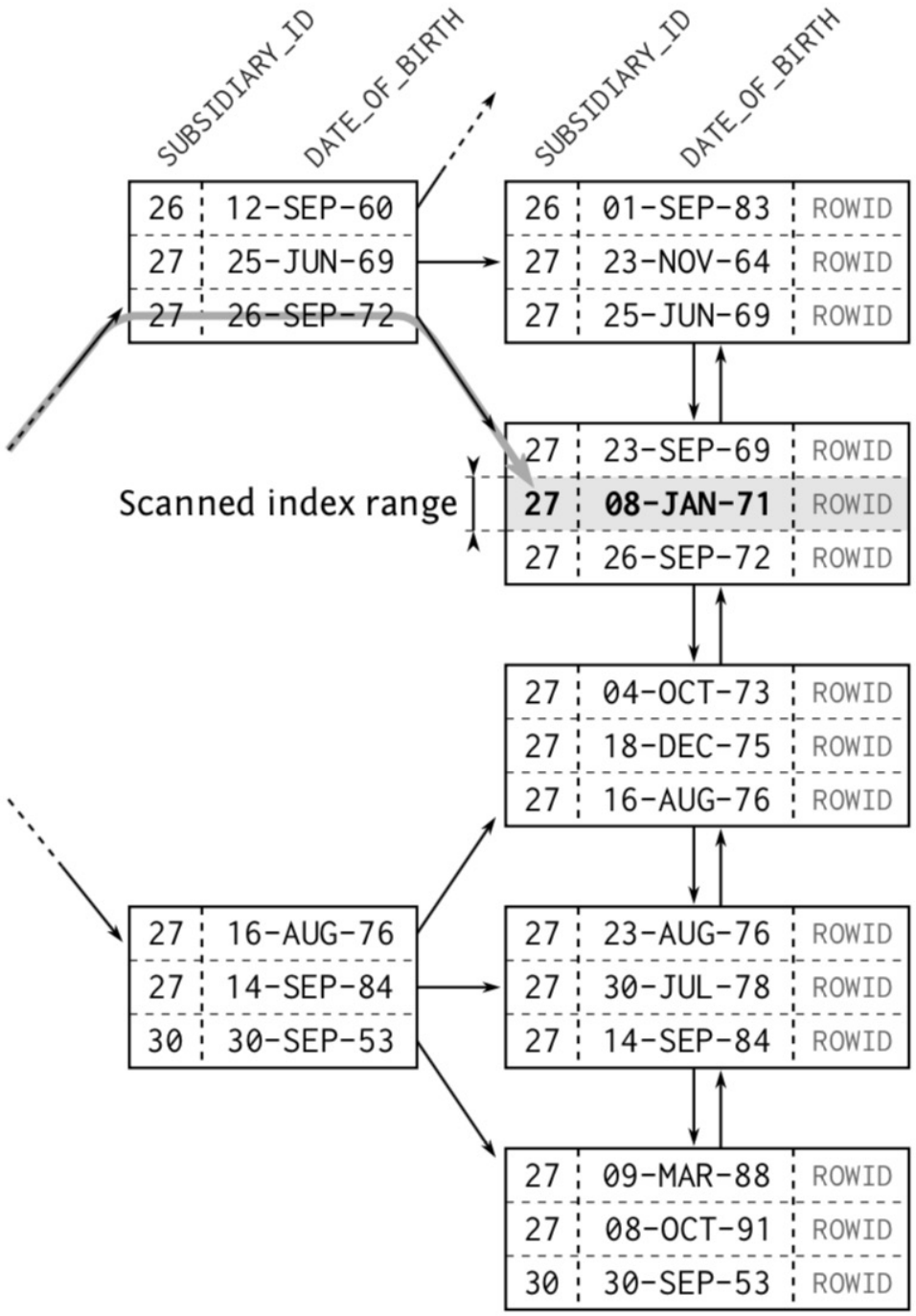
Clustered index

- Tabuľlka je uložená v rámci disku podľa primárného indexu - rýchly sekvenčný sken
  - Napr. MySQL

Ukážky na indexy

- Viac atribútový index
- Indexy a funkcie
  - napr. case-insensitive
- Index only scan
  - nazývany aj covering index
  - agregačné query su vhodný kandidát na tento typ indexov

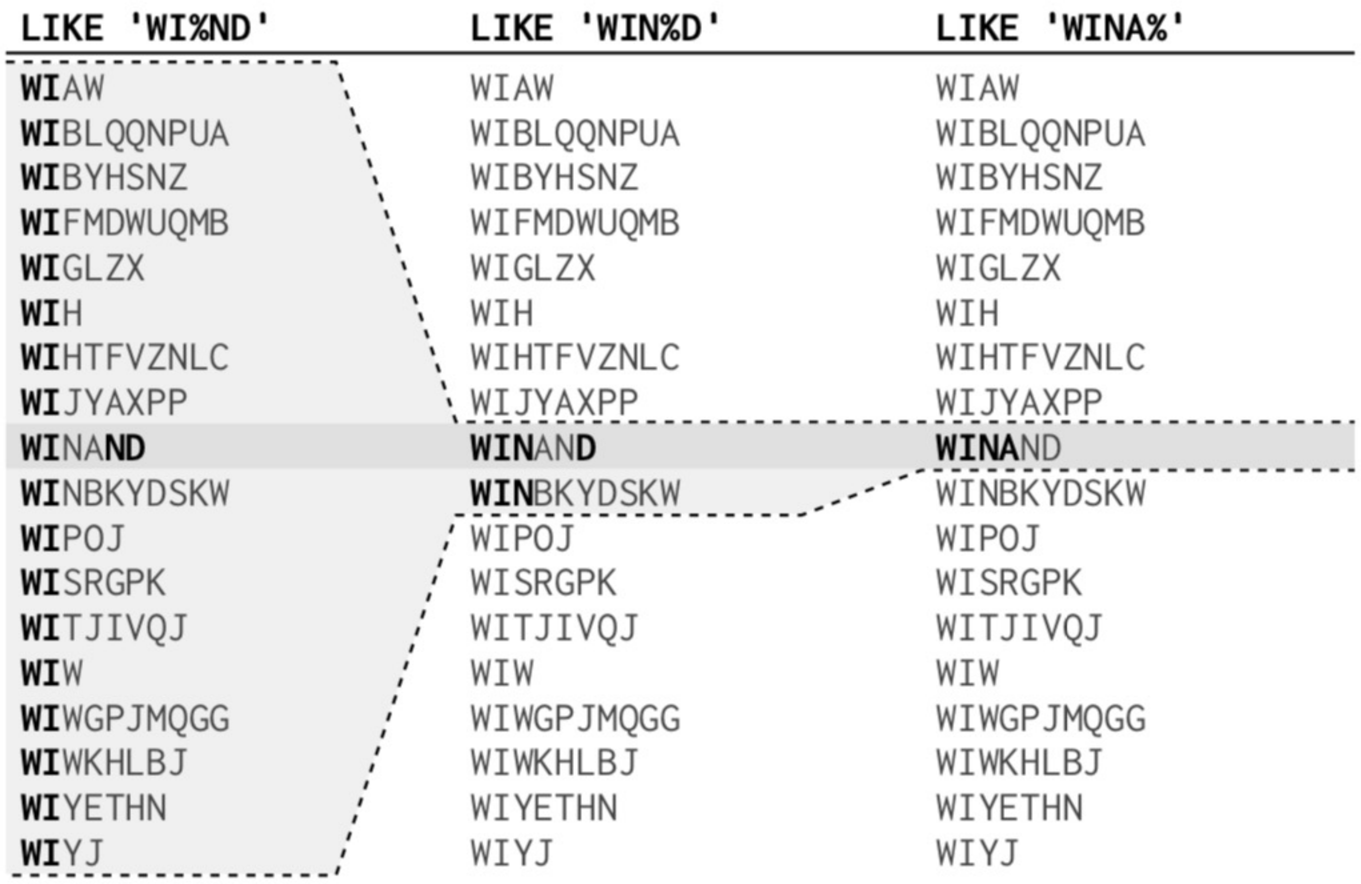

DATE_OF_BIRTH
SUBSIDIARY_ID
DATE_OF_BIRTH
SUBSIDIARY_ID
27-DEC-70 19
01-JAN-71 6
05-JAN-71 3
28-DEC-70 4 ROWID
01-JAN-71 3 ROWID
01-JAN-71 6 ROWID
02-JAN-71 1 ROWID
04-JAN-71 1 ROWID
05-JAN-71 3 ROWID
06-JAN-71 4 ROWID
06-JAN-71 11 ROWID
08-JAN-71 6 ROWID
08-JAN-71 6
09-JAN-71 17
12-JAN-71 3
08-JAN-71 27 ROWID
09-JAN-71 10 ROWID
09-JAN-71 17 ROWID
09-JAN-71 17 ROWID
09-JAN-71 30 ROWID
12-JAN-71 3 ROWID
Scanned index range


SUBSIDIARY_ID
DATE_OF_BIRTH
SUBSIDIARY_ID
DATE_OF_BIRTH
26 12-SEP-60
27 25-JUN-69
27 26-SEP-72
26 01-SEP-83 ROWID
27 23-NOV-64 ROWID
27 25-JUN-69 ROWID
27 23-SEP-69 ROWID
27 08-JAN-71 ROWID
27 26-SEP-72 ROWID
Scanned index range
27 04-OCT-73 ROWID
27 18-DEC-75 ROWID
27 16-AUG-76 ROWID
27 16-AUG-76
27 14-SEP-84
30 30-SEP-53
27 23-AUG-76 ROWID
27 30-JUL-78 ROWID
27 14-SEP-84 ROWID
27 09-MAR-88 ROWID
27 08-OCT-91 ROWID
30 30-SEP-53 ROWID

Zdroj: [1]

LIKE

- berie sa iba po prvý výskyt znaku reprezentujúci wildcard % alebo _
- V prípade bind parametrov postgreSQL predpokladá použitie wildcard znaku na prvom mieste, takže sa index nepoužije vôbec



Zdroj: [1]

Užitočné linky

- Témy súvisiace s indexovaním a načo je potrebné si dať pozor
  - http://use-the-index-luke.com/
- PostgreSQL – vnútorne fungovanie
  - https://www.interdb.jp/pg/index.html