Viacrozmerné bodové vyhľadávanie

Viacrozmerné bodové vyhľadávanie je vyhľadávanie podľa hodnôt viacerých sekundárnych kľúčov súčasne.

Doteraz sme predpokladali, že v zázname je len jeden kľúč (s unikátnymi hodnotami) - tzv. *primárny* kľúč (mohol sa skladať z viacerých atribútov).
Teda výsledkom operácie hľadania je *jedna* hodnota.
Záznam ale môže obsahovať ďalšie položky (ktoré už nie sú unikátne).
Ak pripustíme vyhľadávanie aj podľa hodnôt týchto záznamov, tak ich nazveme „*sekundárne*“ kľúče.
Výsledkom operácie hľadania podľa sekundárneho kľúča je celá množina záznamov.

V nasledujúcom súbore je záznam s ďalšími položkami (atribútmi):

Primárny kľúč	Sekundárne kľúče					Ostatné info.
ŠPZ	Vlastník	Rok výroby	Typ	Farba	Plánov. prehliadka	
BA 0035	Novák	1926	Škoda	Biela	2009	
BA1136	Bajza	2001	Škoda	Žltá	2012	
BB4838	Doktor	1999	Opel	Biela	2009	

Možné vyhľadávanie:

- **vozidlá ktoré vlastní Bajza**
- **vozidlá s plánovanou prehliadkou v r. 2012**
- **vozidlá typu Opel**

Uvedené dotazy sú *jednoduché* dotazy (argumentom je jediný kľúč).

***Zložené* dotazy používajú *kritérium* dotazu vo forme booleovského výrazu:**

a) **Disjunktívne (typu *OR*) - booleovský výraz s použitím *OR***
   - **(vlastník = Novák *OR* vlastník = Doktor)**
   - **(vlastník = Novák OR typ = Opel)**
b) **Konjunktívne (typu *AND*) - booleovský výraz s použitím *AND***

- **(rok výroby = 1926 *AND* farba = biela)**

c) Zmiešané - s použitím *OR* a *AND*

- **(farba = Biela *OR* farba = Žltá) *AND* (vlastník = Doktor)**

***Bodový* dotaz používa v argumente jednu hodnotu (rok výroby = 1926, rok výroby = 1927, rok výroby = 1928).**
***Intervalový* dotaz používa v argumente interval hodnôt (rok výroby Є < 1926, 1928>).**

Bez špecifických štruktúr je pre vyhľadávanie podľa sekundárnych kľúčov nutné pre každý argument prehliadnuť celú štruktúru - extrémne neefektívne => návrh nových štruktúr.

Viaczoznamová štruktúra (Multilist)

Princíp: k sekundárnym kľúčom sa vybudujú explicitné sekundárne indexy ako mechanizmy pre vyhľadávanie. Typické použitie je pre dáta uložené v súboroch, ale princíp je možné využiť aj pri dátach uložených v operačnej pamäti.

C. Adr	Primárny kľúč	Sekundárne kľúče						Informácie
	ŠPZ	Typ	Ďalší Typ	Farba	Ďalší Farba	Plánov. prehliadka	Ďalší Prehl.	
1	BA 0035	Škoda	2	Biela	.	2009		
2	BA1136	Škoda	7	Žltá	7	2012		
3	BB4838	Opel	6	Biela	1	2009		
4	CA2145	Citroen	11	Žltá	.	2008		
5	HU1122	Fiat	9	Modrá	.	2007		
6	KE2541	Opel	10	Červená	.	2012		
7	KE9612	Skoda	.	Žltá	4	2009		
8	LE0250	Toyota	.	Biela	3	2010		
9	PP5665	Fiat	.	Fialová	.	2008		
10	PR1255	Opel	.	Čierna	.	2020		
11	SE3322	Citroen	.	Biela	8	2007		

Typ	Zoz	Dĺžka
Škoda	1	3
Opel	3	3
Fiat	5	2
Citroen	4	2
Toyota	8	1

Sekundárny index podľa „Typ“

Farba	Zoz	Dĺžka
Žltá	2	3
Fialová	9	1
Červená	6	1
Modrá	5	1
Čierna	10	1
Biela	11	4

Sekundárny index podľa „Farba“

Hodnoty s rovnakým sekundárnym kľúčom sú zreťazené pomocou indexov (prípadne adries) v poliach ĎalšíTyp (ďalší prvok s rovnakým typom) resp. ĎalšíFarba (ďalší prvok s rovnakou farbou). Hlavy týchto zreťazených zoznamov sú v poli Zoz (začiatok zoznamu) v sekundárnych indexoch.
Dĺžka: počet záznamov v podsúbore (podzozname) pre danú hodnotu. Urýchľuje vyhľadávanie pri zložených dotazoch.

Dotazy:

a.) Jednoduchý dotaz: prehliadne sa celý (jeden) podzoznam.

b.) Konjunktívny dotaz (X = x *AND* Y = y)

- **sprístupni index pre X, nájdi adresu $A_1(x)$ a dĺžku d(x) spojené s X = x.**
- **sprístupni index pre Y, nájdi adresu $A_1(y)$ a dĺžku d(y) spojené s Y = y.**
- **ak: d(x) < d(y) prehľadávaj reťazec pre X = x a kontroluj, či Y = y**
- **inak: prehľadávaj reťazec pre Y =y a kontroluj, či X = x**

pri hľadaní bielych Toyot prehliadneme 1 záznam
pri hľadaní žltých Fiatov prehliadneme 2 záznamy

c.) Dizjunktívny dotaz (X = x *OR* Y = y)

- **sprístupni index pre X, nájdi adresu $A_1(x)$ spojenú s X = x.**
- **sprístupni index pre Y, nájdi adresu $A_1(y)$ spojenú s Y = y.**
- **nájdi záznamy prehľadaním oboch reťazcov.**

nájdenie bielych a žltých áut vyžiada sprístupnenie 7 záznamov

Vlastnosti:

- **ide o vybudovanie sekundárnych indexov, každý index je uložený v samostatnom súbore**
- **veľké zefektívnenie oproti neindexovanému súboru**
- **veľká spotreba dát pre adresy v hlavnom súbore i v indexoch**
- **pomerne efektívne konjunktívne dotazy**
- **užívateľ musí *vopred* rozhodnúť, ktoré atribúty budú sekundárnymi kľúčmi a tým dať pevný tvar hlavnému súboru**
- **málo efektívne pre dynamické operácie: pri rušení alebo vkladaní záznamov nutné aktualizovať položky Ďalší v hlavnom súbore, príp. aj sekundárne indexy (záznamy zoznamu sú v rôznych blokoch!).**

Implementácia hlavného súboru: ľubovoľná, vzhľadom na primárny kľúč (je možné použiť napr. B strom,...).
Implementácia indexu: ľubovoľná, vzhľadom na sekundárny kľúč.

Tento princíp je možné ľubovoľne modifikovať. Napríklad môžeme mať vybudované sekundárne indexy uložené v operačnej pamäti (napr. RB strom), ale dáta sú v súbore (napr. dynamický hešovací súbor,...). Taktiež je princíp možné použiť na implementáciu viacrozmerného bodového vyhľadávania na štruktúrach uložených len v operačnej pamäti.

Modifikácia, ktorá môže pri niektorých štruktúrach (napr. lineárny hešovací súbor) zefektívniť dynamické operácie (vlož, vymaž), ale zároveň znižuje efektívnosť vyhľadávania používa namiesto adries primárny kľúč.
Táto modifikácia sa používa častejšie pretože súbor s menším množstvom explicitne vyjadrených vzťahov vedie k ľahšej a bezpečnejšej „údržbe“ pri dynamických súboroch.

C. Adr	Primárny kľúč	Sekundárne kľúče	
	ŠPZ	Typ	Ďalší Typ
1	BA 0035	Škoda	BA1136
2	BA1136	Škoda	KE9612
3	BB4838	Opel	
4	CA2145	Citroen	
5	HU1122	Fiat	
6	KE2541	Opel	
7	KE9612	Skoda	

Typ	Zoz	Dĺžka
Škoda	BA0035	3

Sekundárny index podľa „Typ“

Invertované súbory

- **hlavný súbor je v pôvodnom tvare bez akýchkoľvek doplňujúcich informácií týkajúcich sa prístupu podľa sekundárnych kľúčov**
- **ku každému sekundárnemu kľúču je skonštruovaný index (invertovaný index)**

Ak sú všetky atribúty okrem primárneho kľúča sekundárnymi kľúčmi, je súbor *úplne invertovaný* (každý atribút má invertovaný index), inak je *neúplne* invertovaný.
Záznam invertovaného súboru: (hodnota sekundárneho kľúča, zoznam primárnych kľúčov s touto hodnotou) - pre všetky hodnoty.

C. Adr	Primárny kľúč	Sekundárne kľúče			
	ŠPZ	Typ	Farba	Plánov. prehl	Ostatná info
1	BA 0035	Škoda	Biela	2009	
2	BA1136	Škoda	Žltá	2012	
3	BB4838	Opel	Biela	2009	
4	CA2145	Citroen	Žltá	2008	
5	HU1122	Fiat	Modrá	2007	
6	KE2541	Opel	Červená	2012	
7	KE9612	Škoda	Žltá	2009	
8	LE0250	Toyota	Biela	2010	
9	PP5665	Fiat	Fialová	2008	
10	PR1255	Opel	Čierna	2020	
11	SE3322	Citroen	Biela	2007	

Variant:

Namiesto primárnych kľúčov adresy

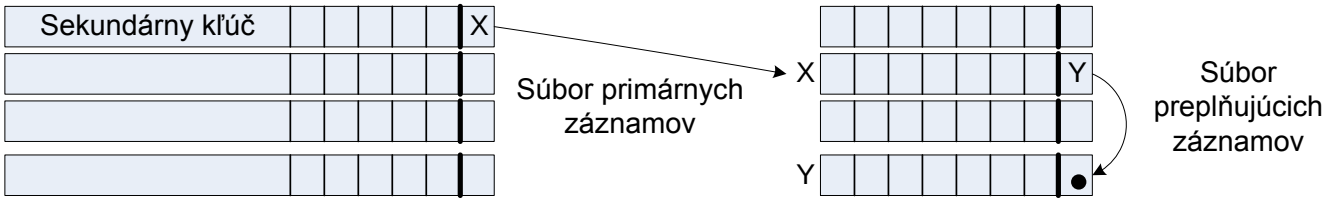
Typ	Primárne kľúče
Škoda	BA0035, BA1136, KE9612
Opel	BB4838, PR1255
Fiat	HU1122, PP5665
Citroen	CA2145, SE3322
Toyota	LE0250

Invertovaný index pre „Typ“

Farba	Primárne kľúče
Žltá	BA1136,CA2145,KE9612
Fialová	PP5665
Červená	KE2541
Modrá	HU1122
Čierna	PR1255
Biela	SE3322,LE0250,BB4838,BA0035

Invertovaný index pre „Farba“

Záznamy s premenlivou dĺžkou. Možná implementácia - ako dva súbory záznamov s pevnou dĺžkou:



a.) Konjunktívny dotaz (X = x *AND* Y = y)

- v invertovanom indexe pre X vyhľadaj všetky primárne kľúče $K_1(x)$, ... $K_n(x)$ spojené s X = x

- v invertovanom indexe pre Y vyhľadaj všetky primárne kľúče $K_1(y)$, ... $K_m(y)$ spojené s Y = y

- nájdi prienik $K(x, y) = K_1(x), ... K_n(x) \cap K_1(y), ... K_m(y)$

- sprístupni záznamy s primárnymi kľúčmi K(x, y) (teda ak K(x, y) je prázdne, netreba ani prehliadať hlavný súbor).

b.) Dizjunktívny dotaz (X = x *OR* Y = y)

- v invertovanom indexe pre X vyhľadaj všetky kľúče $K_1(x), ... K_n(x)$ spojené s X = x

- v invertovanom indexe pre Y vyhľadaj všetky kľúče $K_1(y), ... K_m(y)$ spojené s Y = y

- nájdi zjednotenie $K(x, y) = K_1(x), ... K_n(x) \cup K_1(y), ... K_m(y)$

- sprístupni záznamy s primárnymi kľúčmi K(x, y)

Vlastnosti:

- **efektívnejšia realizácia (hlavne zložitých) dotazov ako v multilist štruktúre**
- **väčšia náročnosť na množstvo dát**
- **zmeny v hlavnom súbore vyvolávajú zmeny v indexoch**
- **štruktúra hlavného súboru nezávislá na indexoch**

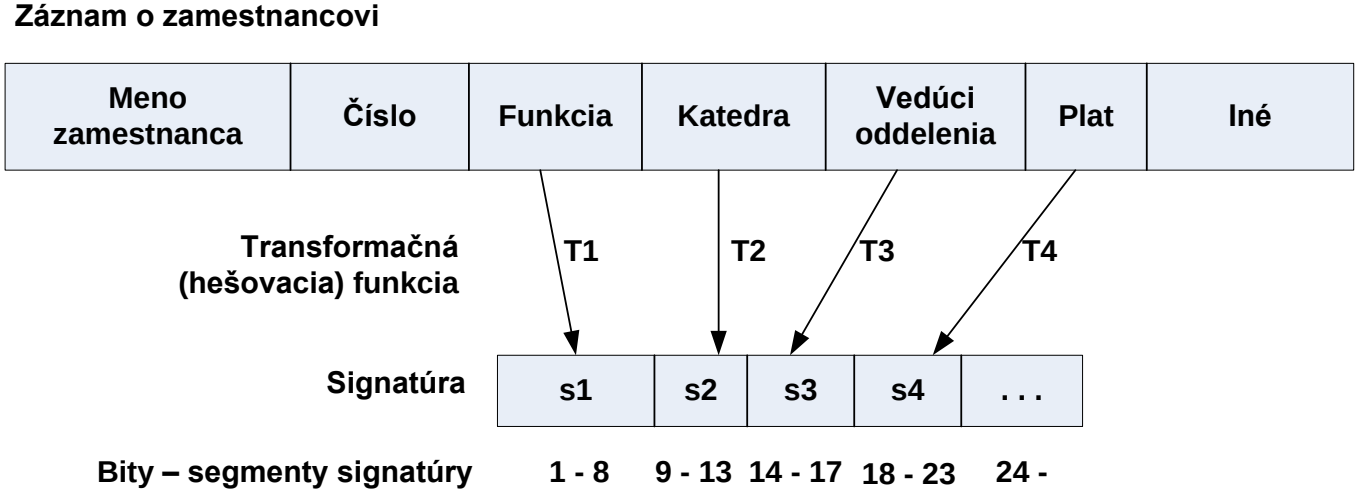
Vyhľadávanie podľa čiastočnej zhody
(partial match retrieval)

<u>Problém</u>: Ako rýchlo vyhľadávať podľa sekundárnych kľúčov (zložené dotazy)
Štruktúra je vhodná na viacrozmerné bodové vyhľadávanie dát uložených v súbore.
<u>Postup</u> (využívajúci princíp hešovania):

- **ku každému záznamu vytvoríme transformáciou sekundárnych kľúčov jeho *signatúru* (deskriptor, „odtlačok prsta“) vo forme jedného bitového poľa,**
- **pri realizácii zloženého dotazu prehľadávame iba *súbor signatúr* a testujeme zhodu (čiastočnú) iba v odpovedajúcich bitoch**
- **ak nájdeme zhodu signatúry, tak záznam vyhľadáme v hlavnom súbore podľa primárneho kľuča.**

Hlavný súbor: ľubovoľná zo známych implementácií.

Tvorba signatúry - príklad:



Typicky: každý sekundárny kľúč má inú hešovaciu funkciu T.
Hešovacia funkcia určí, v ktorom bite *b* príslušného segmentu bude 1.
Vhodné, aby samé nuly znamenali neexistenciu záznamu.

Napríklad:

Funkcia T1 b = **Funkcia** *mod* $2^8 + 1$

Funkcia T2 b =
- **1, ak názov katedry začína na A - E**
- **2, ak názov katedry začína na F - J**
- **3, ak názov katedry začína na K - M**
- **4, ak názov katedry začína na N - R**
- **5, ak názov katedry začína na S - Z**

Funkcia T3 b = **Vedúci oddelenia** *mod* $2^4 + 1$

Funkcia T4 b =
- **1, ak Plat <= 600**
- **2, ak Plat (600, 800>**
- **3, ak Plat (800, 1000>**
- **4, ak Plat (1000, 1150>**
- **5, ak Plat (1150, 1200>**
- **6, ak Plat > 1200**

Príklad takto vytvorených signatúr:

	signatúry segmentov			
s1	**s2**	**s3**	**s4**	**signatúra záznamu**
Asistent	**Elektroniky**	**Klos**	**1090**	
00001000	**00001**	**0100**	**000100**	**000010000000010100000100**
Docent	**Ekonomiky**	**Ežo**	**1000**	
00010000	**00001**	**0001**	**001000**	**000100000000010001001000**
Technik	**Práva**	**Nosák**	**500**	
00000010	**01000**	**1000**	**100000**	**000000100100010001000000**
8 bitov	**5 bitov**	**4 bity**	**6 bitov**	**23 bitov**

Súbor signatúr sa pripraví pri „predspracovaní“ hlavného súboru. Pre každý záznam hlavného súboru sa vytvorí práve jeden záznam v súbore so signatúrami. Záznamy sú tu samozrejme tiež zhlukované do blokov.

Organizácia súboru signatúr:

- ***signatúra + adresa***
- ***signatúra + primárny kľúč.***

Pre rôzne hodnoty sekundárneho kľúča môžu byť vygenerované rovnaké signatúry segmentov (prípad kolízie hešovania) .
V príklade: katedry Ekonomiky a Elektroniky.
Pre zmenšenie pravdepodobnosti kolízie - zväčšiť rozsah segmentu (počet bitov).

Spracovanie konjunktívneho dotazu:

- **vytvoriť „*vyhľadávaciu signatúru*“ pridaním (operáciou „and“) výsledkov hešovania podľa funkcií T pre každý použitý sekundárny kľúč do prázdnej (samé nuly) signatúry**
  - **pre dotaz „docenti s platom (1000, 1150> “ bude vyhľadávacia signatúra 00010000000000000001000**
  - **porovnať vyhľadávaciu signatúru so všetkými signatúrami záznamov, pri zhode príslušných bitov záznamu s vyhľadávacou signatúrou sa porovná vyhľadávací záznam s nájdeným záznamom v hlavnom súbore (kvôli možnosti kolízie), ak nesúhlasí, bola to falošná zhoda signatúr**

Nevýhoda: nutnosť prehliadnuť všetky záznamy signatúr v súbore so signatúrami, ale:

- **možnosť uplatniť rýchle logické binárne operácie počítača,**
- **súbor signatúr je podstatne menší ako hlavný dátový súbor.**

Zlepšená modifikácia vyhľadávania podľa čiastočnej zhody: strom signatúr uložený v súbore

K existujúcemu súboru signatúr sa pripraví ďalší súbor, ktorý bude obsahovať strom signatúr.
Súbor signatúr je rozdelený na bloky (úroveň 1) (v príklade sa používa blokovací faktor f = 5).
Pre každý blok sa do súboru so stromom signatúr (úroveň 2) vloží „*OR* súčet“ všetkých záznamov bloku zo súboru signatúr.
Signatúry na úrovni 2 sa samozrejme uložia do blokov (v príklade sa používa blokovací faktor f = 3). Takto sa postupuje až pokým nevznikne vrchol stromu.
Operácie sú v analógii s B^+ - stromom.
Na obrázku sú znázornené bloky veľkosti 5, záznamy 2 a 18 v hlavnom súbore sú prázdne (napr. z dôvodu uchovania priaznivej hustoty, napr. ak je hlavný súbor statický hešovací súbor).
Výsledkom je menej prehliadnutých záznamov v súbore signatúr ako pri organizácii bez stromu *(v príklade sa do hlavného súboru ide iba raz, lebo signatúry nekolidujú, ináč možno aj viackrát podľa kvality hešovacích funkcií).*
Pri pohybe v strome signatúr sa využíva operácia „and“, ktorá sa aplikuje na vyhľadávanú signatúru a signatúru príslušného podstromu. Ak sa na príslušnom mieste v hľadanej signatúre nachádza 1 a vo výsledku je 0, je isté, že v danej vetve sa hľadaná signatúra nenachádza. Toto umožňuje vylúčiť zo spracovania časť vetiev stromu.
Táto technika je vhodná, keď strom nie je príliš vysoký.
S narastajúcou výškou stromu postupne dochádza k obsadzovaniu všetkých miest signatúry jednotkami, čo je prirodzený dôsledok aplikovania operácie „or“. Signatúra so samými jednotkami neumožňuje v strome rozhodnúť, že sa daná vetva nemá prehľadávať.