
8. Zpracování dotazu

8.1. Podstata optimalizace zpracování dotazu.....	2
8.2. Postup optimalizace zpracování dotazu.....	3
8.2.1. Implementace spojení.....	5
8.2.2. Využití statistik databáze k odhadu ceny dotazu	11
8.3. Optimalizace zpracování dotazu u serveru Oracle	13
Literatura.....	16

8.1. Podstata optimalizace zpracování dotazu

Optimalizace zpracování dotazu znamená nalezení (sub)optimální strategie zpracování dotazu.

- nutnost u relačních systémů
- propracované metody optimalizace
- optimalizaci provádí SŘBD (komponenta zvaná optimalizátor zpracování dotazu). Má dostatek informací (více než programátor), Při změně parametrů databáze stačí zkompilovat znovu dotaz, lze prověřit řadu strategií,

Př) „Které pobočky mají nějakého klienta z Ivančic?”

```
SELECT U.pobocka
```

```
FROM Klient K, Ucet U
```

```
WHERE K.r_cislo = U.r_cislo and K.mesto = 'Ivančice'
```

Procedurální vyjádření jako výraz relační algebry - varianty:

```
((Klient JOIN Ucet) WHERE mesto = 'Ivančice')[pobocka]
```

```
((Klient WHERE mesto = 'Ivančice') JOIN Ucet)[pobocka]
```

```
((Klient JOIN Ucet[r_cislo, pobocka]) WHERE mesto = 'Ivančice')
```



8.2. Postup optimalizace zpracování dotazu

1.) převod do vnitřní reprezentace

((Klient JOIN Ucet) WHERE mesto = 'Ivančice')[pobočka]

2.) nalezení ekvivalentního, ale efektivnějšího výrazu

((Klient WHERE mesto='Ivančice')[r_cislo]) JOIN Ucet)[pobočka]

- některá pravidla (heuristiky):

(A JOIN B) WHERE podm_pro_A AND podm_pro_B =
(A WHERE podm_pro_A) JOIN (B WHERE podm_pro_B)

(A WHERE podm1_pro_A) WHERE podm2_pro_A =
A WHERE podm1_pro_A AND podm2_pro_A

(A [atributy1]) [atributy2] = A [atributy2]

(A [atributy1]) WHERE podm1 = (A WHERE podm1) [atributy1]

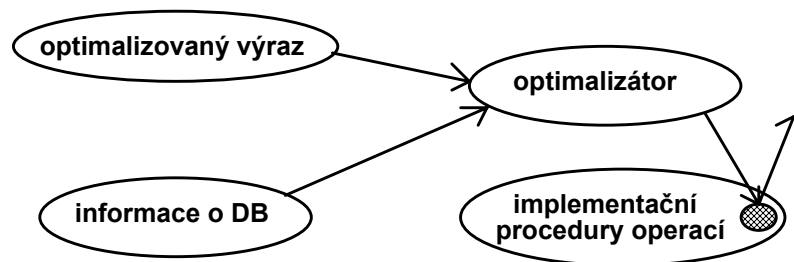
- **sémantická optimalizace** - využití sémantické informace, např. cizí klíče

Př)

(Ucet JOIN Klient) [r_cislo] = Ucet [r_cislo]

3. výběr kandidátů procedur pro implementaci operací

- statistiky databáze, existence indexů, shlukování, ...



4. generování plánu vyhodnocení (access/evaluation/execution plan) a výběr nejlepšího

- cenová funkce: počet diskových operací, případně také čas CPU

8.2.1. Implementace spojení

Př) (Klient JOIN Ucet)

Počet záznamů relace Klient: n_{Klient}

Počet bloků relace Klient: b_{Klient}

Počet záznamů relace Ucet: n_{Ucet}

Počet bloků relace Ucet: b_{Ucet}

- Jednoduchá iterace (zanořené cykly (nested-loops))

```
FOR EACH n-tici K IN Klient DO
  FOR EACH n-tici U IN Ucet DO
    IF K.r_cislo = U.r_cislo THEN
      Zařad' spojené n-tice K a U do výsledku
```

- výpočetní složitost (nejhorší případ):

$n_{Klient} * b_{Ucet} + b_{Klient}$ bloků

(za předpokladu pouze jednoho bloku každé relace ve vyrovnávací paměti)

→ vhodné pro obecné spojení malých relací, resp. když nelze použít efektivnější způsob

- Iterace orientovaná na bloky (block-nested loops)

```
FOR EACH blok dat BKlient OF Klient DO
  FOR EACH blok dat BUcet OF Ucet DO
    FOR EACH n-tici K IN BKlient DO
      FOR EACH n-tici U IN BUcet DO
        IF K.r_cislo = U.r_cislo THEN
          Zařad' spojené n-tice K a U do výsledku
```

- výpočetní složitost:

$b_{Klient} * b_{Ucet} + b_{Klient}$ bloků

- další možné optimalizace (ukončení vnitřního cyklu při rovnosti hodnot a klíči, použití většího prostoru pro bloky vnější relace, využití indexu ve vnitřním cyklu, ...)

- Indexovaná iterace (indexed nested-loops)

- n-tice vnitřní relace se zpřístupňuje využitím indexu

- výpočetní složitost (nejhorší případ):

$b_{Klient} + n_{Klient} * c_{IndexUcet}$ bloků,

kde je cena přístupu $c_{IndexUcet}$ využitím indexu

- **Algoritmus sort/merge**

```
Setříd' soubory záznamů relací Klient a Ucet vzestupně
podle hodnot ve sloupci r_cislo. Výsledkem budou
setříděné relace Klients a Ucets;
K := první n-tice relace Klients;
U' := první n-tice relace Ucets;
WHILE není konec relace Klients nebo relace Ucets DO
/* posbírání n-tic relace Ucets se stejnou hodnotou */
  Vyprazdni pomocnou relaci Stejne;
  U := U' ;
  Vlož do Stejne n-tici U;
  U' := další n-tice relace Ucets;
  WHILE (není konec relace Ucets) AND
    (U.r_cislo = U'.r_cislo) DO
  BEGIN
    Vlož do Stejne n-tici U' ;
    U' := další n-tice relace Ucets;
  END
END
```

```
/* nalezení prvního kandidáta na spojení v Klients */
WHILE (není konec relace Klients) AND
  (K.r_cislo < U.r_cislo) DO
  K := další n-tice relace Klients;
/* zpracování kandidátů v relaci Klients */
WHILE (není konec relace Klients) AND
  (K.r_cislo = U.r_cislo) DO
  BEGIN
    FOR EACH n-tici S IN Stejne DO
      Zařad' spojené n-tice K a U do výsledku;
    K := další n-tice relace Klients;
  END
END
```

- výpočetní složitost:

$b_{KlientS} + b_{UcetS}$ bloků setříděných souborů (vejde-li pomocná relace Stejne do paměti) + doba setřídění

Klients

r_cislo	jmeno	
440726/0672	Jan Novák	
510230/048	Pavel Tomek	
530610/4532	Petr Veselý	
580807/9638	Josef Mádr	
601001/2218	Ivan Zeman	
625622/6249	Jana Malá	

Ucets

c_uctu	r_cislo	
4320286	440726/0672	
2075752	440726/0672	
1182648	530610/4532	
3564852	580807/9638	
2001793	580807/9638	
5853961	601001/2218	
1582549	625622/6249	

- Hašované spojení

```
/* Rozčlenění */
```

```
FOR EACH n-tici K IN Klient DO
```

```
    Zařad' n-tici do sekce  $H_{K_i}$ , kde  $i = h(K.r\_cislo)$ 
```

```
FOR EACH n-tici U IN Ucet DO
```

```
    Zařad' n-tici do sekce  $H_{U_i}$ , kde  $i = h(U.r\_cislo)$ 
```

```
/* Spojování po sekcích metodou indexované iterace*/
```

```
FOR i = 0 TO  $n_i$  DO
```

```
BEGIN
```

```
    Vytvoř hašovaný index nad  $H_{U_i}$  pro r_cislo;
```

```
    FOR EACH n-tici K IN  $H_{K_i}$  DO
```

```
        BEGIN
```

```
            Použitím hašovaného indexu nad  $H_{U_i}$  najdi všechny  
            n-tice  $U_j$  relace Ucet se stejnou hodnotou  
            atributu r_cislo;
```

```
            FOR EACH souhlasnou n-tici U IN  $H_{U_i}$  DO
```

```
                Zařad' spojené n-tice K a U do výsledku;
```

```
        END
```

```
    END
```

8.2.2. Využití statistik databáze k odhadu ceny dotazu

- Základní používané statistiky

n_R ... počet n-tic v relaci R ,

s_R ... velikost n-tice relace R

$V(X, R)$... počet různých hodnot atributu X v relaci R

Př) $R(A), X \in A \quad R \text{ WHERE } X = c \dots$ odhad velikosti

$n_R/V(X, R)$ - selektivita

$n_R/V(X, R) * s_R$ - odhad velikosti výsledné relace

Selektivita atributu X relace R je průměrný počet n-tic připadajících na jednu hodnotu atributu X v relaci R .

Př) $R(A), S(B) \quad RS = R \text{ JOIN } S$

Kolik n-tic bude mít relace RS ?

Varianty:

$A \cap B = \{\}$ $\Rightarrow n_{RS} = n_R * n_S$

$X = A \cap B$ je PK relace $R \Rightarrow n_{RS} = n_S$

$X = A \cap B \neq \{\}$ a není PK $\Rightarrow n_{RS} = \min\{ n_R * n_S / V(X, S), n_S * n_R / V(X, R) \}$

- Aktualizace statistik

- provádí administrátor po výraznějších změnách v databázi, při poklesu výkonnosti apod.

8.3. Optimalizace zpracování dotazu u serveru Oracle

- Nastavení režimu optimalizace

```
ALTER SESSION
SET OPTIMIZER_MODE = {FIRST_ROWS | FIRST_ROWS_n
                      | ALL_ROWS | RULE | CHOOSE }
```

Lze zvolit:

- cíl optimalizace

- minimální doba odezvy,
- maximální propustnost

- přístup k optimalizaci

- založený na statistikách - doporučený
- založený pouze na pravidlech

- Zjištění výsledného plánu vyhodnocení

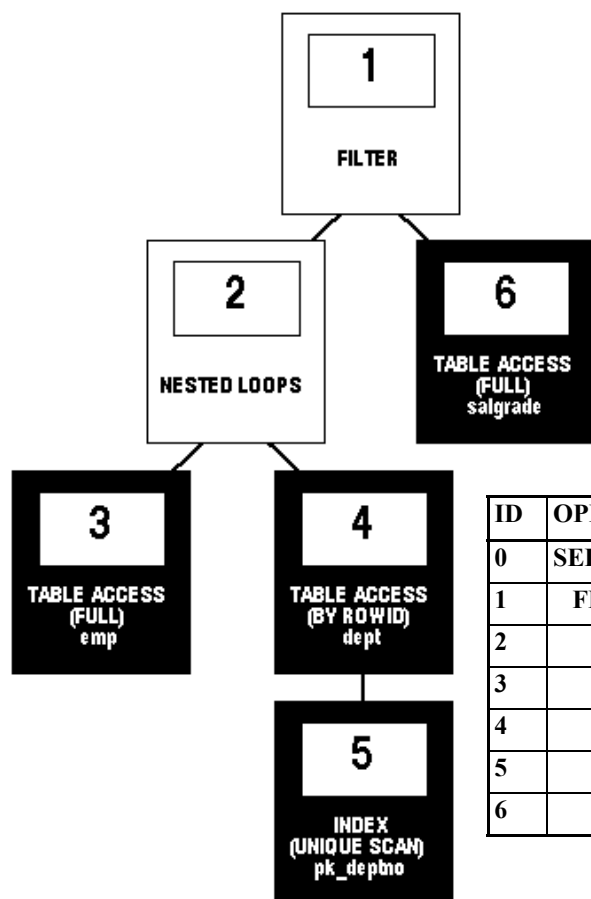
- lze zjistit příkazem EXPLAIN PLAN

- Aktualizace statistik

- použitím balíku DBMS_STATS nebo příkazem ANALYZE

Př) Najdi zaměstnance, jejichž plat nespadá do žádného platového pásma (v tabulce SALGRADE)?

```
SELECT ename, job, sal, dname
FROM EMP, DEPT
WHERE EMP.deptno = DEPT.deptno
AND NOT EXISTS
  (SELECT *
   FROM SALGRADE
   WHERE EMP.sal BETWEEN losal AND hisal);
```



ID	OPERATION	OPTIONS	OBJECT_NAME
0	SELECT STATEMENT		
1	FILTER		
2	NESTED LOOPS		
3	TABLE ACCESS	FULL	EMP
4	TABLE ACCESS	BY ROWID	DEPT
5	INDEX	UNIQUE SCAN	PK_DEPTNO
6	TABLE ACCESS	FULL	SALGRADE

Literatura

1. Silberschatz, A., Korth H.F, Sudarshan, S.: Database System Concepts. Fourth Edition. McGRAW-HILL. 2001, str. 493 – 564.
2. Date C.J.: An Introduction to Database Systems. Sixth edition. Addison-Wesley, 1995, str. 501 – 542.
3. Pokorný, J.: Databázová abeceda. Science, Veletiny, 1998, str. , 23 – 26, 77 – 80, 119 – 124.
4. Oracle9i Database Performance Tuning Guide and Reference. Oracle Corp. March 2002, str. 1-1 – 9-22.