Operátor Cube v SQL

slajdy k přednášce DJ1

Jaroslav Pokorný

MFF UK, Praha

pokorny@ksi.mff.cuni.cz

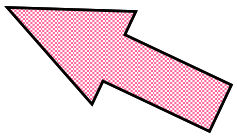
Obsah

1. Motivace pro operátor CUBE
   - Nedostatečné možnosti GROUP BY
   - Jak dělat agregace
2. Operátor CUBE a ROLLUP
3. Využití
4. Závěry

OLAP

- OLAP (Online Analytical Processing)
- Princip modelování: dimenze, fakty
  - dimenze
    - mohou být hierarchické
    - mají atributy
  - fakty
    - atributy závislé na dimenzích
- Př.: analýza prodeje aut
  - zajímáme se o vliv modelu, barvy a roku výroby auta a nezajímá nás dealer ani datum prodeje

OLAP

- n-rozměrné struktury dat
- možnosti reprezentace:
  - jedna tabulka pro všechno
  - tabulka pro každou dimenzi + tabulka faktů
  - datová kostka 
- výpočty:
  - agregační funkce COUNT, SUM, MAX, ...
  - operátor GROUP BY

Problémy s GROUP BY

- Př.: Prodej aut

  Dimenze: Model, Rok, Barva

  Fakty: Počet prodaných kusů
- Snadno: běžné agregace jako

```
SELECT Model, Barva, SUM(Počet)
FROM Prodeje
GROUP BY Model, Barva;
```

- Spíše složitě: Který model se nejlépe prodával na Slovensku?
- Omezení na budování agregačních konstrukcí jako jsou:
  - histogramy
  - roll-up
  - křížové tabulky

Histogramy

- V SQL není běžně umožněna přímá konstrukce histogramů

  Př.: Počasí(čas, z_šířka, z_výška, teplota)

  máme každodenní hlášení o počasí, chtěli bychom umět sdružovat dni do týdnů či měsíců a pro každé území chceme minimální měsíční teplotu.
- Histogramy jsou vypočítávané pomocí zahnízděných dotazů

Histogramy

Moderní SQL systémy podporují histogramy přímo (není potřeba používat hnízděné dotazy jako v SQL92)

```
SELECT měsíc, území, MIN(teplota)
FROM Počasí
GROUP BY Měsíc(čas) AS měsíc,
                Území(z_šířka,z_výška) AS území
```

Roll-up, drill-down

- data můžeme agregovat do různých úrovní dimenzí
- pohyb po úrovních

  směrem nahoru: roll-up,

  směrem dolů: drill-down

Podle: Model, Rok, Barva

Podle: Model, Rok

Podle: Model

Model	Rok	Barva			
Chevy	1994	Černá	50		
		Bílá	40		
				90	
	1995	Černá	85		
		Bílá	115		
				200	
					290

Kam s agregovanými hodnotami?

- Nevýhody předchozí reprezentace:
  - prázdné hodnoty v řádcích
  - není to relace
  - enormní množství atributů (domén)
- Dílčí řešení:
  - agregované hodnoty je vhodné mít přímo v tabulce
  - např. předáme sloupce, které ukazují hodnoty agregačních funkcí pro každý řádek
  - Nevýhoda: mimo RMD

Model	Rok / Barva						Celkem
	1994		Celkem	1995		Celkem	
	Černá	Bílá		Černá	Bílá		
Chevy	50	40	90	85	115	200	290
Ford	50	10	60	85	75	160	220
Celkem	100	50	150	170	190	360	510

Kam s agregovanými hodnotami?

Řešení: relační reprezentace

- speciální hodnota ALL
- ALL vyjadřuje, že na daném místě tabulky chceme všecky hodnoty z dané domény.
- ALL() definuje množinu

  Př.: ALL(Barva) = {Černá, Bílá}

Model	Rok	Barva	Kusů
Chevy	1994	Černá	50
Chevy	1994	Bílá	40
Chevy	1994	ALL	90
Chevy	1995	Černá	85
Chevy	1995	Bílá	115
Chevy	1995	ALL	200
Chevy	ALL	ALL	290

Jak s Roll-up v SQL?

```
SELECT ‘ALL’, ‘ALL’, ‘ALL’, SUM(Kusů)
    FROM Prodeje
    WHERE Model=‘Chevy’
UNION
SELECT Model, ‘ALL’, ‘ALL’, SUM(Kusů)
    FROM Prodeje
    WHERE Model=‘Chevy’
    GROUP BY Model
UNION
SELECT Model, Rok, ‘ALL’, SUM(Kusů)
    FROM Prodeje
    WHERE Model=‘Chevy’
    GROUP BY Model, Rok
UNION ...
```

- Nebo několik příkazů SELECT bez ALL

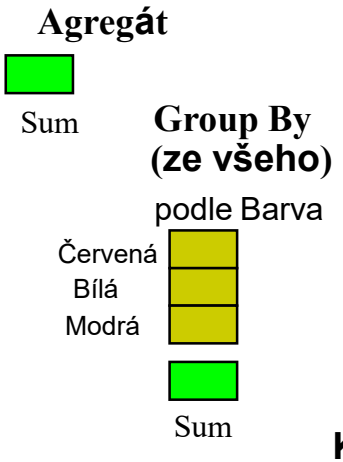
Křížová tabulka

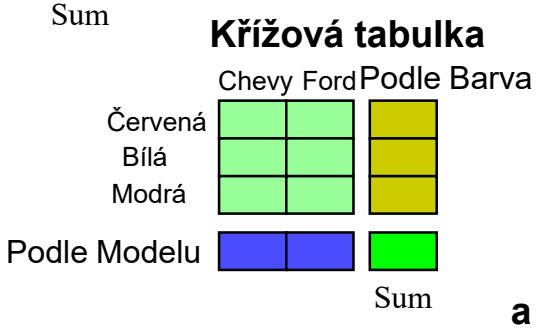
- Mírně pozměníme relační reprezentaci, a dostáváme křížovou tabulku
  - hodnoty dimenze jsou záhlaví řádků a sloupců v dvourozměrném prostoru
- Pomocí SQL: GROUP BY + UNION
- Problém: jak pro Ford? Další tabulka.

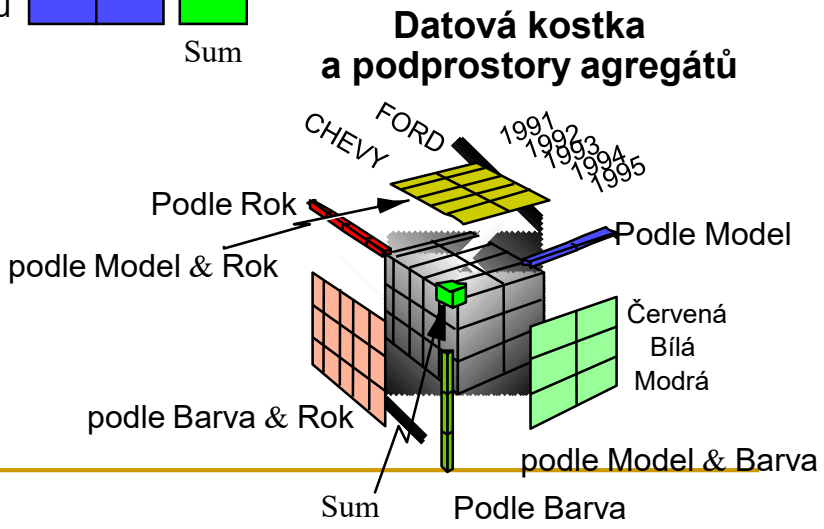
		rok		
	Chevy	1994	1995	Celkový počet (ALL)
barva	Černá	50	85	135
	Bílá	40	115	155
	Celkový počet (ALL)	90	200	290

Operátory CUBE a ROLLUP

- řešení: operátory ROLLUP a CUBE
- zobecnění operátoru GROUP BY, resp. křížové tabulky







CUBE - prvotní představa

- Př.: tvoříme datovou kostku ze tří atributů
  - výsledek připomíná skutečnou 3D kostku *C*
  - hrany *C* představují domény atributů, obsahem buněk jsou hodnoty - fakty.
  - každá buňka odpovídá jedné SQL skupině (grupě)
  - pro každý atribut „nalepíme“ na okraj *C* agregovanou hodnotu, vzniklou použitím GROUP BY v jednom rozměru
  - na hrany *C* vycházející z „počátku“ „nalepíme“ hodnoty vypočítané přes dva atributy (redukci dimenzí o úroveň výše)
  - V „počátku“*C* se nachází „superagregát“ - hodnota vypočítaná přes všechny hodnoty nacházející se v *C*
- Datová kostka je vícedimenzionální model dat, kde každá doména atributu obsahuje speciální hodnotu ALL.

CUBE – jak funguje

- Operátor CUBE funguje velmi podobně:
  - je ekvivalentní kolekci standardních GROUP BY pro *všechny* podmnožiny specifikovaných atributů (seskupení),
  - k výsledku přidává jednotlivé (super)agregáty,
  - ponechává i původní řádky (seskupení s prázdnou podmnožinou specifikovaných atributů).
- přibylo: při N atributech 2^N-1 typů agregovaných hodnot
- jestliže $C_i = |dom(A_i)|$, $i \in <1,N>$, potom je velikost kostky $\Pi(C_i + 1)$.
- pomocí CUBE se fakta a agregace zpracovávají pro všechny buňky nejednou
- Pz.: výhody speciální implementace - např. při pokusech na MS SQL Server 2005 byl CUBE 2x rychlejší než GROUP BY a UNION

Syntaxe

- připomeňme syntaxi GROUP BY:

  GROUP BY <seznam_atributů_k_agregaci>
  <seznam_atributů_k_agregaci> ::=
  {(<jméno_sloupce> | <výraz>)
  [AS <jméno>]
  ,...}

- seznam atributů k agregaci je speciálním případem seskupení

Redukce počtu seskupení

- Někdy je vybudování celé kostky zbytečné
- Někdy jsou všecky možné kombinace atributů zbytečné, nepotřebujeme je, či jsou nesmyslné (např. aplikace CUBE na atributy den, měsíc, rok)
  - GROUPING SETS – seskupení výčtem
  - ROLLUP vrací pouze hodnoty pro hierarchii atributů, kterou specifikujeme

GROUPING SETS

Př.: Prodej aut

Dimenze: Model, Země, Barva

Fakty: Počet prodaných kusů

- explicitní zadání seskupení

```
SELECT Model, Barva, Země, SUM(Kusů)
FROM Prodeje
GROUP BY GROUPING SETS ((),(Model), (Barva,
    Země))
```

Operátor ROLLUP

- operátor ROLLUP je „úspornější“, vyprodukuje jen následující agregáty:

  (v_1, v_2 , ... , v_k , f()),
  (v_1, v_2 , ... , ALL, f()),
  ...
  (v_1, ALL , ... , ALL, f()),
  (ALL, ALL, ..., ALL, f())

- Podmnožiny z 1. atributem ALL se nedostanou do výsledku (vyjma řádku se superagregátem)
  - Výsledky jsou kompaktnější
  - Nelze použít pro všechny dotazy, jako u CUBE
    (D.: „Kolik se prodalo bílých aut?“ .... Ne!)

CUBE operátor

```
SELECT agr_kusů = SUM(Kusů),
       Model, Země, Barva
FROM Prodeje
GROUP BY CUBE
(Model, Země, Barva);
```

Model	Země	Barva	Kusů
Chevy	CZ	Bílá	45
Chevy	CZ	Žlutá	18
Chevy	CZ	Černá	78
Chevy	SK	Bílá	41
Chevy	SK	Žlutá	52
Chevy	SK	Černá	61
Ford	CZ	Bílá	28
Ford	CZ	Žlutá	47
Ford	CZ	Černá	30
Ford	SK	Bílá	21
Ford	SK	Žlutá	46
Ford	SK	Černá	8

CUBE

Agr_kusů	Model	Země	Barva
45	Chevy	CZ	Bílá
18	Chevy	CZ	Žlutá
78	Chevy	CZ	Černá
141	Chevy	CZ	ALL
41	Chevy	SK	Bílá
52	Chevy	SK	Žlutá
61	Chevy	SK	Černá
154	Chevy	SK	ALL
295	Chevy	ALL	ALL
28	Ford	CZ	Bílá
47	Ford	CZ	Žlutá
30	Ford	CZ	Černá
105	Ford	CZ	ALL
21	Ford	SK	Bílá
46	Ford	SK	Žlutá
8	Ford	SK	Černá
75	Ford	SK	ALL

36 řádků

180	Ford	ALL	ALL
475	ALL	ALL	ALL
73	ALL	CZ	Bílá
65	ALL	CZ	Žlutá
108	ALL	CZ	Černá
246	ALL	CZ	ALL
62	ALL	SK	Bílá
98	ALL	SK	Žlutá
69	ALL	SK	Černá
229	ALL	SK	ALL
86	Chevy	ALL	Bílá
49	Ford	ALL	Bílá
135	ALL	ALL	Bílá
70	Chevy	ALL	Žlutá
93	Ford	ALL	Žlutá
163	ALL	ALL	Žlutá
139	Chevy	ALL	Černá
38	Ford	ALL	Černá
177	ALL	ALL	Černá

CUBE operátor

```
SELECT agr_kusů = SUM(Kusů),
       Model, Země, Barva
FROM Prodeje
GROUP BY ROLLUP
(Model, Země, Barva);
```

Model	Země	Barva	Kusů
Chevy	CZ	Bílá	45
Chevy	CZ	Žlutá	18
Chevy	CZ	Černá	78
Chevy	SK	Bílá	41
Chevy	SK	Žlutá	52
Chevy	SK	Černá	61
Ford	CZ	Bílá	28
Ford	CZ	Žlutá	47
Ford	CZ	Černá	30
Ford	SK	Bílá	21
Ford	SK	Žlutá	46
Ford	SK	Černá	8

CUBE

Agr_kusů	Model	Země	Barva
45	Chevy	CZ	Bílá
18	Chevy	CZ	Žlutá
78	Chevy	CZ	Černá
141	Chevy	CZ	ALL
41	Chevy	SK	Bílá
52	Chevy	SK	Žlutá
61	Chevy	SK	Černá
154	Chevy	SK	ALL
295	Chevy	ALL	ALL
28	Ford	CZ	Bílá
47	Ford	CZ	Žlutá
30	Ford	CZ	Černá
105	Ford	CZ	ALL
21	Ford	SK	Bílá
46	Ford	SK	Žlutá
8	Ford	SK	Černá
75	Ford	SK	ALL

19 řádků

180	Ford	ALL	ALL
475	ALL	ALL	ALL
73	ALL	CZ	Bílá
65	ALL	CZ	Žlutá
108	ALL	CZ	Černá
246	ALL	CZ	ALL
62	ALL	SK	Bílá
98	ALL	SK	Žlutá
69	ALL	SK	Černá
229	ALL	SK	ALL
86	Chevy	ALL	Bílá
49	Ford	ALL	Bílá
135	ALL	ALL	Bílá
70	Chevy	ALL	Žlutá
93	Ford	ALL	Žlutá
163	ALL	ALL	Žlutá
139	Chevy	ALL	Černá
38	Ford	ALL	Černá
177	ALL	ALL	Černá

Vztahy GROUP BY, CUBE, ROLLUP

- Platí (algebraické zákony):
  - CUBE(ROLLUP) = CUBE
  - CUBE(GROUP BY) = CUBE
  - ROLLUP(GROUP BY) = ROLLUP
- $\Rightarrow$ smysl má hierarchie těchto 3 operátorů:

GROUP BY <seznam_atr_k_agregaci>

ROLLUP <seznam_atr_k_agregaci>

CUBE <seznam_atr_k_agregaci>

Syntaxe

Přechod ke CUBE a ROLLUP:

GROUP BY [<seznam_atr_k_agregaci>]

[ROLLUP <seznam_atr_k_agregaci>]

[CUBE <seznam_atr_k_agregaci>]

- Pz.: někdy jiná syntaxe, např. MS SQL Server 2005

  GROUP BY <seznam_atr_k_agregaci> WITH {CUBE | ROLLUP}

Více typů agregací

- za GROUP BY lze použít více ROLLUP a CUBE
- každý generuje seznamy atributů k agregaci (seskupení); celkově se uvažuje jejich kartézský součin

Př.:

```
SELECT Model, Barva, Země, SUM(Kusů)
FROM Prodeje
GROUP BY ROLLUP (Model), ROLLUP(Barva, Země)
```

generuje seskupení

{Model, ()} X {(Barva, Země), (Barva), ()}

= { (Model, Barva, Země), (Model, Barva), (Model), (Barva, Země), (Barva), ( ) }

Hodnota ALL

- problémy s ALL jako zvláštní hodnotou:
  - mnoho speciálních případů
  - jestliže ALL reprezentuje množinu, potom ostatní hodnoty z domény této množiny musí být jednoduché typy
- přistoupilo se proto k následující implementaci hodnoty ALL:
  - místo hodnoty ALL se použije hodnota NULL
  - neimplementuje se funkce ALL()
  - implementuje se funkce GROUPING() na rozlišování hodnot NULL a ALL

Hodnota ALL

- dříve: hodnota ALL
- nyní: v datové oblasti hodnota NULL
- v odpovídajícím poli hodnota TRUE vyjadřující, že dané NULL má význam ALL
- dříve: (ALL, ALL, ALL, 941)
- nyní: (NULL,NULL,NULL,941,TRUE,TRUE,TRUE)

GROUPING

- NULL suplující ALL nazýváme seskupovací (grouping NULL)
- Funkce GROUPING odlišuje význam seskupovacího NULL od normálního NULL
  - vrací 1, jestliže jde o seskupovací NULL (ALL)
  - vrací 0, jestliže jde o NULL nebo je na daném místě ne-NULL hodnota.

GROUPING

- Pak lze psát:

```
SELECT Model, Rok, Barva, SUM(Kusů),
                          GROUPING(Model),
                          GROUPING(Rok),
                          GROUPING(Barva)
FROM Prodeje
GROUP BY CUBE Model, Rok, Barva.
```

GROUPING()

- INSERT INTO Prodeje

  VALUES (NULL, ‘SK’, NULL, 229);
  - Tento nový řádek je nerozlišitelný od jiného řádku při použití CUBE
  - rozliší je pouze funkce GROUPING()

GROUPING()

Model	Země	Barva	Kusů
NULL	SK	NULL	229
Chevy	CZ	Bílá	45
Chevy	CZ	Žlutá	18
Chevy	CZ	Černá	78
Chevy	SK	Bílá	41
Chevy	SK	Žlutá	52
Chevy	SK	Černá	61
Ford	CZ	Bílá	28
Ford	CZ	Žlutá	47
Ford	CZ	Černá	30
Ford	SK	Bílá	21
Ford	SK	Žlutá	46
Ford	SK	Černá	8

```
SELECT Agr_kusů = SUM(Kusů),
       Model, Barva, Země
FROM Prodeje
GROUP BY Model, Barva, Země
WITH CUBE;
```

45	Chevy	Bílá	CZ
41	Chevy	Bílá	SK
86	Chevy	Bílá	**NULL**
....			
229	**NULL**	**NULL**	SK
....			
229	**NULL**	**NULL**	SK

ALL ..... Grouping(Model) = 0

NULL ..... Grouping(Model) = 1

GROUPING()

```
SELECT Agr_kusů = SUM(Kusů),
   Model,
   'všechny_modely'=grouping(Model),
   Země,
   'všechny_země'=grouping(Země),
   Barva,
   'všechny_barvy'=grouping(Barva)
FROM Prodeje
GROUP BY CUBE Model, Barva, Země;
```

Model	Země	Barva	Kusů
NULL	SK	NULL	229
Chevy	CZ	Bílá	45
Chevy	CZ	Žlutá	18
Chevy	CZ	Černá	78
Chevy	SK	Bílá	41
Chevy	SK	Žlutá	52
Chevy	SK	Černá	61
Ford	CZ	Bílá	28
Ford	CZ	Žlutá	47
Ford	CZ	Černá	30
Ford	SK	Bílá	21
Ford	SK	Žlutá	46
Ford	SK	Černá	8

GROUPING()

Agr_kusů	Model	Všechny _modely	Země	Všechny _země	Barva	Všechny _barvy
45	Chevy	0	CZ	0	Bílá	0
41	Chevy	0	SK	0	Bílá	0
86	Chevy	0	NULL	1	Bílá	0
....						
229	NULL	0	SK	0	NULL	0
....						
229	NULL	1	SK	0	NULL	1

Operátor Cube - shrnutí

- Operátor CUBE zobecňuje a sjednocuje:
  - agregáty
  - group by
  - roll-up and drill-down
  - křížové tabulky
- Zajímavé problémy:
  - počítání CUBE pro různé agregační funkce
  - implementace (hašování, 2^N algoritmus, CUBE algoritmus)

Závěr

- Operátor CUBE a ROLLUP jsou standardizovány v SQL:1999.
- Vytváření datové kostky vyžaduje speciální implementaci.
- Styl dotazování: omezení prostoru specializovaným dotazem (WHERE) a pak CUBE
- Další vývoj v praxi: zejména Microsoft – MDX (**M**ulti**D**imensional E**X**pressions)