



Hurtownie danych - przegląd technologii

Robert Wrembel
Politechnika Poznańska
Instytut Informatyki

Robert.Wrembel@cs.put.poznan.pl
www.cs.put.poznan.pl/rwrembel



Efektywność OLAP

- **Perspektywy zmaterializowane**
 - przepisywanie zapytań
 - wybór zbioru perspektyw
 - anomalie odświeżania
- **Optymalizacja GROUP BY**
- **Kompresja**
- **Przetwarzanie równoległe**
- **Partycjonowanie**



Przepisywanie zapytań - przykład (1)

```
create materialized view mv_sprzedaz
build immediate
refresh force
next sysdate + (1/24)
ENABLE QUERY REWRITE
as
select sk.miasto, pr.prod_nazwa, cz.nazwa_miesiaca,
       sum(sp.l_sztuk) sprzedano, sum(sp.wartosc) wartosc
from sprzedaz sp, sklepy sk, produkty pr, czas cz
where sp.sklep_id=sk.sklep_id
and sp.produkt_id=pr.produkt_id
and sp.data=cz.data
group by sk.miasto, pr.prod_nazwa,
         cz.nazwa_miesiaca;
```

```
select sk.miasto, pr.prod_nazwa,
       sum(sp.wartosc) wartosc
from sprzedaz sp, sklepy sk,
     produkty pr, czas cz
where sp.sklep_id=sk.sklep_id
and sp.produkt_id=pr.produkt_id
and sp.data=cz.data
and sk.miasto='Poznań'
group by sk.miasto, pr.prod_nazwa,
         cz.nazwa_miesiaca;
```

Execution Plan

```
-----
0      SELECT STATEMENT Optimizer=CHOOSE (Cost=2 Card=1 Bytes=37)
1      0      TABLE ACCESS (FULL) OF 'MV_SPRZEDAZ' (Cost=2 Card=1 Bytes=37)
```



Przepisywanie zapytań - przykład (2)

- W celu wyznaczenia wyników zapytania użytkownika, perspektywa zmaterializowana jest łączona z jedną z jej tabel bazowych
- Perspektywa musi zawierać albo klucz podstawowy tej tabeli bazowej lub ROWID rekordów tabeli bazowej

```
create materialized view mv_sprzedaz1
build immediate
refresh complete
enable query rewrite
as
select sk.miasto, pr.produkt_id, cz.nazwa_miesiaca,
       sum(sp.wartosc) as wartosc
from sprzedaz sp, sklepy sk, produkty pr, czas cz
where sp.sklep_id=sk.sklep_id
and sp.produkt_id=pr.produkt_id
and sp.data=cz.data
group by sk.miasto, pr.produkt_id, cz.nazwa_miesiaca;
```



Przepisywanie zapytań - przykład (3)

```
select sk.miasto, pr.prod_nazwa, sum(sp.wartosc) wartosc
from sprzedaz sp, sklepy sk, produkty pr, czas cz
where sp.sklep_id=sk.sklep_id
and sp.produkt_id=pr.produkt_id
and sp.data=cz.data
group by sk.miasto, pr.prod_nazwa, cz.nazwa_miesiaca;
```

- zależność funkcyjna **PRODUKT_ID → PROD_NAZWA**
 - wyznaczona na podstawie klucza



Perspektywa zmaterializowana

- ⇒ Wybór właściwego zbioru perspektyw zmaterializowanych
- ⇒ Kryteria
 - minimalizacja czasu odpowiedzi jak największej liczby zapytań
 - minimalizacja czasu odpowiedzi najbardziej kosztownych zapytań
 - minimalizacja kosztów odświeżania perspektyw
 - minimalizacja wykorzystania przestrzeni dyskowej
- ⇒ Problem trudny
- ⇒ Wsparcie ze strony oprogramowania systemowego
 - Oracle9i/10g/11g - Summary Advisor/Access Advisor



Algorytm zachłanny

⇒ Założenia

- znany jest szacowany rozmiar danych dla każdego zapytania
- koszt wykonania zapytania jest reprezentowany rozmiarem odczytanych danych

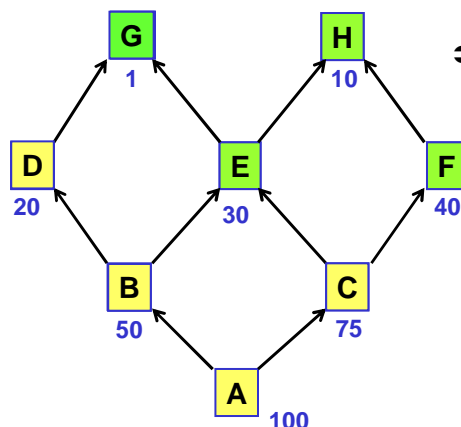
⇒ Cel

- minimalizacja rozmiaru danych odczytywanych przez zapytania przy zadanej liczbie zmaterializowanych perspektyw

- ⇒ Harinarayan V., Rajaraman A., Ullman J.: Implementing Data Cubes Efficiently. SIGMOD, 1996



Przykład działania



⇒ Węzeł ⇒ zapytanie

- koszt wykonania zapytania ⇒ rozmiar odczytanych danych

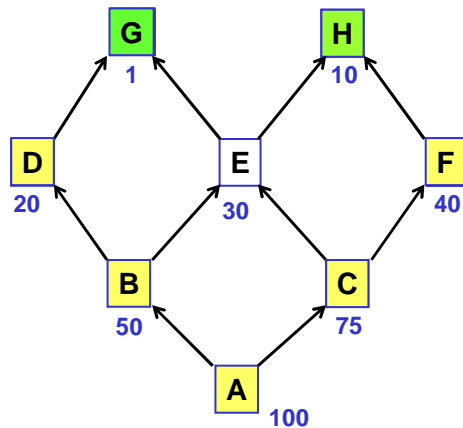
⇒ Łuk ⇒ możliwość wyliczenia zapytania (węzeł docelowy łuku) na podstawie innego zapytania (węzeł początkowy łuku)

⇒ Przebieg 1

- B: $50 \times 5 = 250$
- C: $25 \times 5 = 125$
- D: $80 \times 2 = 160$
- E: $70 \times 3 = 210$
- F: $60 \times 2 = 120$
- G: 99×1
- H: 90×1



Przykład działania

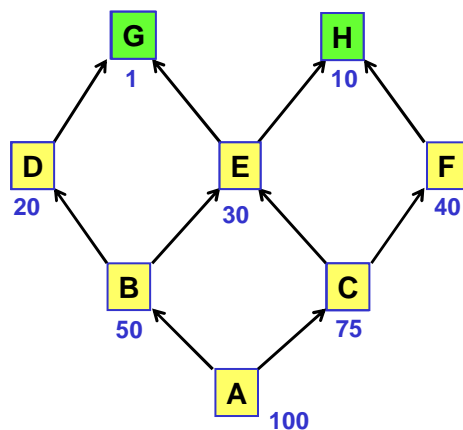


⇒ Przebieg 2

- C: $25 \times 2 = 50$ (C i F, H z B)
- D: $30 \times 2 = 60$
- E: $20 \times 3 = 60$
- F: $60 + 10$ (B-F) = 70
- G: 49
- H: 40



Przykład działania



⇒ Przebieg 3

- C: $25 = 25$ (C, E z B)
- D: $30 \times 2 = 60$
- E: $20 + 20 + 10$ (E-F) = 50
- G: 49
- H: 30



Perspektywa zmaterializowana

- ⇒ **Konieczność odświeżania**
 - automatycznie z zadaną częstotliwością/na żądanie
 - przyrostowo/całkowicie
- ⇒ **Wykrywanie zmian w źródłach**
 - analiza dziennika transakcji (transaction, redo log)
 - wyzwalacze
 - porównanie dwóch stanów źródła
 - własna implementacja dziennika
- ⇒ **Anomalie odświeżania**
 - równoczesna praca transakcji modyfikujących zawartość źródeł danych i procesów odświeżania



Anomalie odświeżania

R1		R2		V=R1 join R2		
A	B	B	C	A	B	C
1	2	2	4	1	2	4
4	2			4	2	4
				4	2	4

1. Wstawienie rekordu [2,4] do R2 ⇒ źródło informuje HD komunikatem $U1 = \text{insert}(R2, [2,4])$
2. HD konstruuje zapytanie odświeżające $Q1 = R1 \text{ join } [2,4]$
3. Wstawienie rekordu [4,2] do R1 ⇒ źródło informuje HD komunikatem $U2 = \text{insert}(R1, [4,2])$
4. HD konstruuje zapytanie odświeżające $Q2 = [4,2] \text{ join } R2$
5. Źródło wykonuje zapytanie Q1 i przekazuje wynik $W1 = \{[1,2,4], [4,2,4]\}$
6. Źródło wykonuje zapytanie Q2 i przekazuje wynik $W2 = \{[4,2,4]\}$



Techniki unikania anomalii

⇒ Kompensacja

- Zhuge Y., Garcia-Molina H., Hammer J., Widom J.: View Maintenance in Warehousing Environment. SIGMOD, 1995
- Zhuge Y., Garcia-Molina H., Wiener J.: The Strobe Algorithms for Multi-Source Warehouse Consistency. PDIS, 1996

⇒ Systemowe wersje danych

- Quass D., Widom J.: On-line Warehouse View Maintenance. SIGMOD, 1997
- Teschke M., Ulbrich A.: Concurrent Warehouse Maintenance Without Compromising Session Consistency. DEXA, 1998
- Kulkarni S., Mohania M.: Concurrent Maintenance of Views Using Multiple Versions. IDEAS, 1999



Kompensacja

⇒ System przechowuje

- kolejkę zapytań KZ w trakcie wykonywania
- tymczasową tabelę WZ z wynikami dotychczas wykonanych zapytań

R1		R2	
A	B	B	C
1	2	2	4

1. Wstawienie rekordu [2,4] do R2 ⇒ źródło informuje HD komunikatem $U1 = \text{insert}(R2, [2,4])$
2. HD konstruuje zapytanie odświeżające $Q1 = R1 \text{ join } [2,4]$
3. Q1 jest wstawiane do KZ, $KZ = \{Q1\}$



Kompensacja

R1		R2	
A	B	B	C
1	2	2	4
4	2		

3. Wstawienie rekordu [4,2] do R1 \Rightarrow źródło informuje HD komunikatem $U2 = \text{insert}(R1, [4,2])$
4. $KZ = \{Q1\}$, wynik Q1 obarczony błędem spowodowanym przez $U2 \Rightarrow$ do zapytania Q2 należy dołączyć część kompensującą: $Q2 = [4,2] \text{ join } R2 - [4,2] \text{ join } [2,4]$
5. Q2 jest wstawiane do KZ, $KZ = \{Q1, Q2\}$
6. Źródło wykonuje zapytanie Q1 i przekazuje wynik do WZ, $WZ = \{[1,2,4], [4,2,4]\}$, Q1 jest usuwane z KZ, $KZ = \{Q2\}$
7. Źródło wykonuje zapytanie Q2: $\{[4,2,4]\} - \{[4,2,4]\} = \emptyset$ i przekazuje wynik do WZ, $WZ = \{[1,2,4], [4,2,4]\} + \emptyset$



Systemowe wersje danych

- ⇒ Jedna transakcja odświeżająca HD
- ⇒ Wiele transakcji odczytujących dane (transakcje OLAP)
- ⇒ Algorytm 2VL (Two Version Locking)
 - w HD mogą istnieć pary wersji
 - bieżąca (dla transakcji OLAP) i przyszła (odświeżana)
 - przeszła (dla transakcji OLAP) i bieżąca (po zakończonym odświeżaniu)
 - każdy rekord został rozszerzony o atrybuty przechowujące drugą wersję danych (horyzontalna redundancja)



Systemowe wersje danych

⇒ Redundancja wertykalna

- kolejne wersje danych przechowywane jako odrębne rekordy
- rozszerzenie schematu perspektywy zmaterializowanej o atrybuty przechowujące znaczniki czasowe początku i końca ważności rekordu
- transakcja OLAP czyta tylko te rekordy, których wartość znacznika czasowego początku ważności jest mniejsza niż znacznik czasowy rozpoczęcia transakcji OLAP



Optymalizacja GROUP BY

transID	sklep	produkt	data	kwota_sprz
1	Żabka1	łaciate 2%	01.02.2008	3,50
2	Żabka1	kajzerka	01.02.2008	2,20
3	Żabka1	kajzerka	01.02.2008	5,50
4	Żabka2	Polityka	02.02.2008	4,50
5	Żabka2	masło ekstra	02.02.2008	4,30
6	Żabka2	Polityka	02.02.2008	4,50

```
select sklep, produkt, data, sum(kwota_sprz)
from sprzedaz
group by sklep, produkt, data;
```

SKLEP	PRODUKT	DATA	SUM(KWOTA_SPRZ)
-----	-----	-----	-----
Żabka1	łaciate 2%	01.02.2008	3.5
Żabka2	Polityka	02.02.2008	9
Żabka1	kajzerka	01.02.2008	7.7
Żabka2	masło ekstra	02.02.2008	4.3



Optymalizacja GROUP BY

⇒ GROUP BY ROLLUP (sklep, produkt, data)

- GROUP BY sklep, produkt, data
- GROUP BY sklep, produkt
- GROUP BY sklep
- GROUP BY ()

```
select sklep, produkt, data, sum(kwota_sprz)
from sprzedaz
group by ROLLUP(sklep, produkt, data);
```

SKLEP	PRODUKT	DATA	SUM(KWOTA_SPRZ)
Żabka1	kajzerka	01.02.2008	7.7
Żabka1	kajzerka		7.7
Żabka1	łaciate 2%	01.02.2008	3.5
Żabka1	łaciate 2%		3.5
Żabka1			11.2
Żabka2	Polityka	02.02.2008	9
Żabka2	Polityka		9
Żabka2	masło ekstra	02.02.2008	4.3
Żabka2	masło ekstra		4.3
Żabka2			13.3
			24.5



Optymalizacja GROUP BY

⇒ GROUP BY CUBE (sklep, produkt, data)

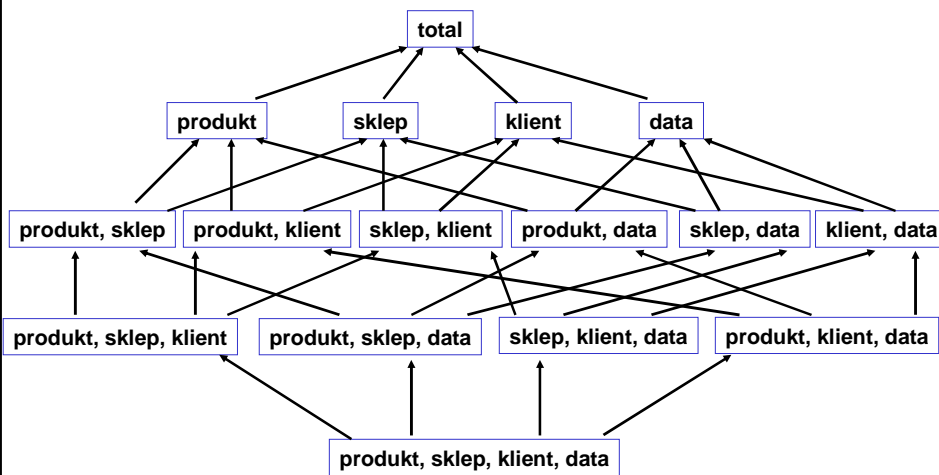
- GROUP BY sklep, produkt, data
- GROUP BY sklep, produkt
- GROUP BY sklep, data
- GROUP BY produkt, data
- GROUP BY sklep
- GROUP BY produkt
- GROUP BY data
- GROUP BY ()

⇒ GROUP BY GROUPING SETS ((sklep, produkt), sklep, ())

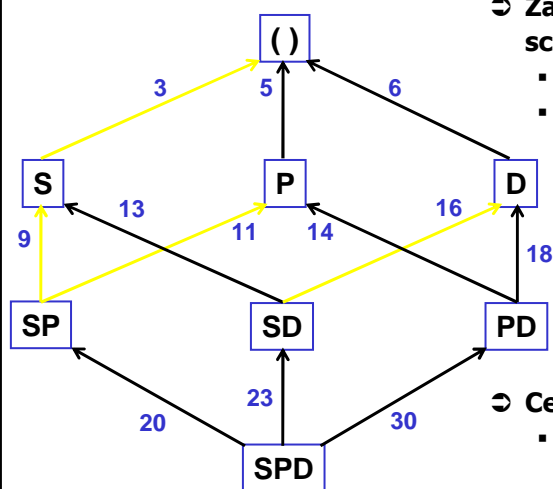
- GROUP BY sklep, produkt
- GROUP BY sklep
- GROUP BY ()



W jaki sposób wyliczać grupowania?



Optymalizacja GROUP BY



- ⇒ Założenia: dla każdego schematu grupowania znana
- liczba unikalnych wartości
 - koszt wyliczenia na podstawie węzła podrzędnego

- ⇒ Cel optymalizacji
- wyznaczyć wszystkie schematy grupowania w sposób minimalizujący sumę kosztów



Techniki optymalizacji

- ⇒ **Najmniejszy węzeł rodzica (smallest parent)**
 - obliczenie grupowania wyższego poziomu na podstawie najmniejszego (rozmiarowo) grupowania niższego poziomu
- ⇒ **Buforowanie wyników w RAM (cache results)**
 - wyniki grupowania są wykorzystane do następnego grupowania
 - **group by SPD -> group by SP -> group by S**
- ⇒ **Ograniczenie odczytów z dysku (amortize scans)**
 - w czasie jednego odczytu z dysku wyliczanych jest wiele schematów grupowania
 - np. jeśli wynik grupowania SPD jest składowany na dysku to wylicza się SP, SD, PD

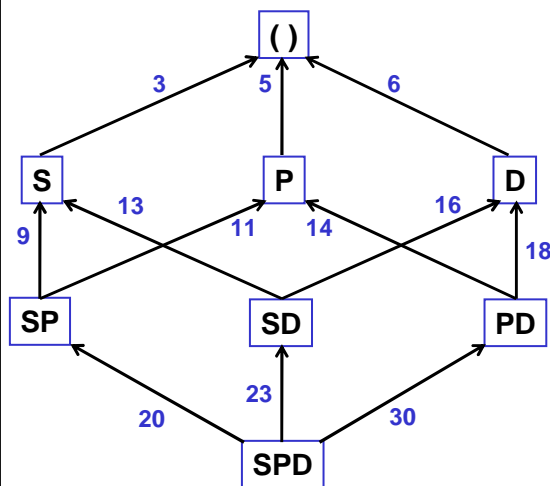


Techniki optymalizacji

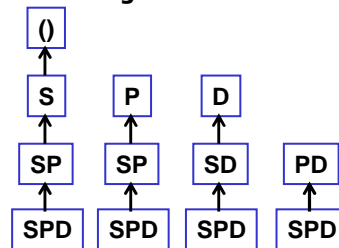
- ⇒ **Współdzielenie wyników sortowania (share sorts)**
 - jeden schemat sortowania jest wykorzystany do wyliczania wielu schematów grupowania
 - ⇒ **Podział dużej tabeli na części (partycje)**
 - zastosowanie haszowania do każdej części
 - scalanie części (możliwe dla funkcji addytywnych)
 - współdzielenie części dla wielu schematów grupowania
- ⇒ Agarwal S., Agrawal R., Deshpande M. P., Gupta A., Naughton F. J., Ramakrishnan R., Sarawagi S.: On the Computation of Multidimensional Aggregates. VLDB, 1996
- ⇒ Ross K. A., Srivastava D.: Fast Computation of Sparse Datacubes. VLDB, 1997
- ⇒ Beyer K, Ramakrishnan R.: Bottom-Up Computation of Sparse and Iceberg Cubes. SIGMOD, 1999
- ⇒ Wang W., Feng J., Lu H., Xu Yu J.: Condensed Cube: An Effective Approach to Reducing Data Cube Size. ICDE, 2002
- ⇒ Lakshmanan L., Pei J., Han J.: Quotient cube: How to summarize the semantics of a data cube. VLDB, 2002
- ⇒ Chen Z., Narasayya V.: Efficient Computation of Multiple Group By Queries. SIGMOD, 2005
- ⇒ Morfonios K., Ioannidis Y.: CURE for Cubes: Cubing Using a ROLAP Engine. VLDB, 2006



Algorytm PipeSort



- ⇒ Przeszukiwanie grafu ↓
- ⇒ Dla każdego poziomu znajdowana jest krawędź o najmniejszym koszcie
- ⇒ Wykonywanie grupowania ↑
- ⇒ Buforowanie wyników grupowania z poziomu niższego



Algorytm PipeHash

- ⇒ Koncepcja jak PipeSort
- ⇒ Założenie: znane jest oszacowanie rozmiaru wierzchołka grafu (wyniku schematu grupowania)
- ⇒ Zamiast sortowania wykorzystywane haszowanie

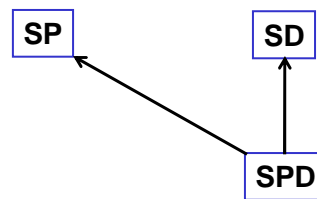


Algorytm Overlap

- ⇒ **Koncepcja jak PipeSort**
- ⇒ **Założenie: znane jest oszacowanie rozmiaru wierzchołka grafu (wyniku schematu grupowania)**
- ⇒ **Przeглядanie grafu w szerz na każdym poziomie**
- ⇒ **Wynik grupowania jest dzielony na części**

Żabka1	łaciate 2%	3.50
Żabka1	kajzerka	7.70

Żabka2	Polityka	9.00
Żabka2	łaciate 2%	3.50
Żabka2	kajzerka	7.70
Żabka2	Polityka	9.00



Algorytmy Memory-Cube i Partitioned-Cube

- ⇒ **Memory-Cube zakłada, że dane zmieszczą się w RAM i sortowanie będzie możliwe w RAM**
- ⇒ **Partitioned-Cube zakłada, że dane nie zmieszczą się w RAM**
 - **podział danych na grupy ze względu na atrybuty grupujące, np. sklep, produkt, data**
 - **dla każdej unikalnej wartości atrybutu wiodącego, np. sklep in {Żabka1, Żabka2, ...} w agregacie na poziomie k**
 - **oblicz agregaty na poziomach k+i za pomocą Memory-Cube**



Algoritmy optymalizacji

Algorytm	Przeglądanie przestrzeni grupowań	Kolejność wykonywania grupowań	Metoda (sortowanie/haszowanie)	Obsługiwane operatory	Hierarchie
PipeSort	↓	↕	Sortowanie	ROLLUP CUBE	TAK
PipeHash	↓	↕	Haszowanie	ROLLUP CUBE	TAK
Overlap	↓	↔	Sortowanie	ROLLUP CUBE	
Partitioned-Cube	↓	↕	Sortowanie	CUBE	
Memory-Cube	↓	↕	Sortowanie	CUBE	
BUC	↑	↕	Sortowanie	CUBE	
BU-BST	↑	↕	Sortowanie	CUBE	
QC-Tables	↑		Sortowanie	CUBE	NIE
Chen - Narasayya	↑	↕		GROUPING SETS	
CURE	↑	↕	Sortowanie Haszowanie	ROLLUP CUBE	TAK



Kompresja danych

➤ Kompresja danych w bloku

blok danych przed kompresją

Eternity Calvin Klein	100
Polo Ralph Laurent	230
Polo Ralph Laurent	90
Polo Ralph Laurent	110
Eternity Calvin Klein	210
Eternity Calvin Klein	340
Eternity Calvin Klein	75

blok danych po kompresji

Eternity Calvin Klein	
Polo Ralph Laurent	
●	100
●	230
●	90
●	110
●	210
●	340
●	75



Kompresja słownikowa

⇒ Słownik kompresji

- może być przechowywany w
 - dedykowanej tabeli lub
 - w nagłówku bloku danych

Kosmetyk	Data	Sklep	Kwota
Polo Ralph Laurent	20-05-06	Sefora	160
Eternity Calvin Klein	21-05-06	Sefora	300
Polo Ralph Laurent	22-05-06	Douglas	170
Eternity Calvin Klein	23-05-06	Sefora	165
Eternity Calvin Klein	23-06-06	Douglas	180

słownik kompresji

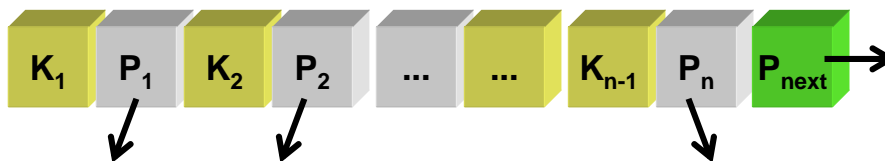
Wartość	Kod
Polo Ralph Laurent	01
Eternity Calvin Klein	02
Sefora	03
Douglas	04

Kosmetyk	Data	Sklep	Kwota
01	20-05-06	03	160
02	21-05-06	03	300
01	22-05-06	04	170
02	23-05-06	03	165
02	23-06-06	04	180

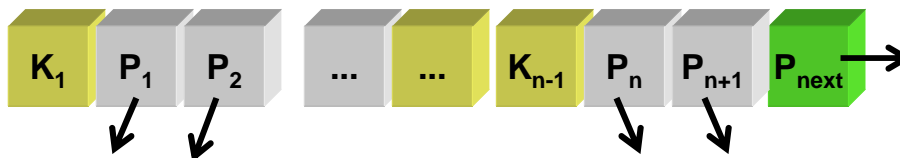


Kompresja indeksów

⇒ Nieskompresowane liście B-drzewa



⇒ Skompresowane liście B-drzewa



⇒ Kompresja map bitowych w indeksach bitmapowych



Przetwarzanie równoległe

⇒ Wykonywanie zapytań

```
SELECT /*+ PARALLEL(sp, 5) */ sklep_id, sum(ilosc)
FROM sprzedaz sp
GROUP BY sklep_id;
```

⇒ Tworzenie kopii tabel

```
create table sklepy_kopia parallel 3
as select * from sklepy;
```

⇒ Budowanie indeksów

⇒ Wczytywanie danych do hurtowni

⇒ Archiwizowanie danych

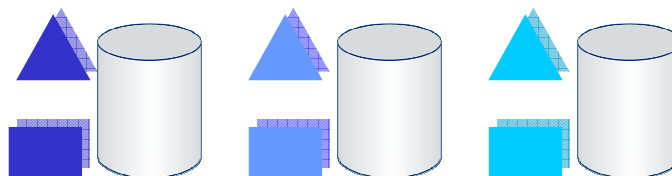
⇒ Odtwarzanie bazy danych po awarii



Partycjonowanie

⇒ Podział tabeli/indeksu na mniejsze fragmenty

- równoległy odczyt partycji
- równoległe wykonywanie zapytania
- zrównoważenie obciążenia dysków
- adresowanie mniejszego zbioru danych (1/n tabeli)





Standardowa organizacja rekordowa

Spółka	Rok	Miesiąc	Dzień	Otw	Min	Max	Zamkn
BZWBK	2006	Mar	31	148,00	147,00	148,00	148,00
BZWBK	2006	Mar	30	149,00	147,50	150,50	148,00
BZWBK	2006	Mar	29	148,50	146,50	148,50	147,50
BZWBK	2006	Mar	28	150,00	148,00	150,00	149,00
BZWBK	2006	Mar	27	147,50	147,50	150,00	150,00
BZWBK	2006	Mar	24	148,00	142,00	150,00	146,00
BZWBK	2006	Mar	23	148,00	147,50	150,00	147,50
BZWBK	2006	Mar	22	147,00	145,50	150,00	147,50
BZWBK	2006	Mar	21	148,50	147,00	150,00	147,00
BZWBK	2006	Mar	20	148,50	147,00	151,50	150,00
BZWBK	2006	Mar	17	151,50	148,00	152,50	148,50
BZWBK	2006	Mar	16	150,00	149,50	152,50	150,00
BZWBK	2006	Mar	15	151,50	149,00	152,00	149,00
BZWBK	2006	Mar	14	149,00	148,50	151,50	150,00
BZWBK	2006	Mar	13	152,50	146,00	152,50	150,00
BZWBK	2006	Mar	10	152,00	146,00	154,50	150,50
BZWBK	2006	Mar	9	154,50	154,00	156,50	154,00
BZWBK	2006	Mar	8	164,50	153,50	165,00	153,50
BZWBK	2006	Mar	7	172,50	165,00	172,50	165,00
BZWBK	2006	Mar	6	170,00	168,00	173,00	173,00
BZWBK	2006	Mar	5	169,50	163,50	171,50	170,00
BZWBK	2006	Mar	2	170,50	168,00	171,50	169,50
BZWBK	2006	Mar	1	166,00	165,00	173,50	171,00

BZWBK	2006	Mar	31	148,00	147,00	148,00	148,00	0,00
BZWBK	2006	Mar	30	149,00	147,50	150,50	148,00	0,34
BZWBK	2006	Mar	29	148,50	146,50	148,50	147,50	-1,01
BZWBK	2006	Mar	28	150,00	148,00	150,00	149,00	-0,67

BZWBK	2006	Mar	27	147,50	147,50	150,00	150,00	2,74
BZWBK	2006	Mar	24	148,00	142,00	150,00	146,00	-1,02
BZWBK	2006	Mar	23	148,00	147,50	150,00	147,50	0,00
BZWBK	2006	Mar	22	147,00	145,50	150,00	147,50	0,34

BZWBK	2006	Mar	21	148,50	147,00	150,00	147,00	-2,00
BZWBK	2006	Mar	20	148,50	147,00	151,50	150,00	1,01
BZWBK	2006	Mar	17	151,50	148,00	152,50	148,50	-1,00
BZWBK	2006	Mar	16	150,00	149,50	152,50	150,00	0,67

BZWBK	2006	Mar	15	151,50	149,00	152,00	149,00	-0,67
BZWBK	2006	Mar	14	149,00	148,50	151,50	150,00	0,00
BZWBK	2006	Mar	13	152,50	146,00	152,50	150,00	-0,33
BZWBK	2006	Mar	10	152,00	146,00	154,50	150,50	-2,27

bloki bazy danych

⇒ Identyfikacja rekordu na podstawie ROWID



Organizacja kolumnowa

Spółka	Rok	Miesiąc	Dzień	Otw	Min	Max	Zamkn
BZWBK	2006	Mar	31	148,00	147,00	148,00	148,00
BZWBK	2006	Mar	30	149,00	147,50	150,50	148,00
BZWBK	2006	Mar	29	148,50	146,50	148,50	147,50
BZWBK	2006	Mar	28	150,00	148,00	150,00	149,00
BZWBK	2006	Mar	27	147,50	147,50	150,00	150,00
BZWBK	2006	Mar	24	148,00	142,00	150,00	146,00
BZWBK	2006	Mar	23	148,00	147,50	150,00	147,50
BZWBK	2006	Mar	22	147,00	145,50	150,00	147,50
BZWBK	2006	Mar	21	148,50	147,00	150,00	147,00
BZWBK	2006	Mar	20	148,50	147,00	151,50	150,00
BZWBK	2006	Mar	17	151,50	148,00	152,00	148,50
BZWBK	2006	Mar	16	150,00	149,50	152,50	150,00
BZWBK	2006	Mar	15	151,50	149,00	152,00	149,00
BZWBK	2006	Mar	14	149,00	148,50	151,50	150,00
BZWBK	2006	Mar	13	152,50	146,00	152,50	150,00
BZWBK	2006	Mar	10	152,00	146,00	154,50	150,50
BZWBK	2006	Mar	9	154,50	154,00	156,50	154,00
BZWBK	2006	Mar	8	164,50	153,50	165,00	153,50
BZWBK	2006	Mar	7	172,50	165,00	172,50	165,00
BZWBK	2006	Mar	6	170,00	168,00	173,00	173,00
BZWBK	2006	Mar	5	169,50	163,50	171,50	170,00
BZWBK	2006	Mar	2	170,50	168,00	171,50	169,50
BZWBK	2006	Mar	1	166,00	165,00	173,50	171,00

⇒ Identyfikacja rekordu na podstawie pozycji (numeru szczeliny) w bloku

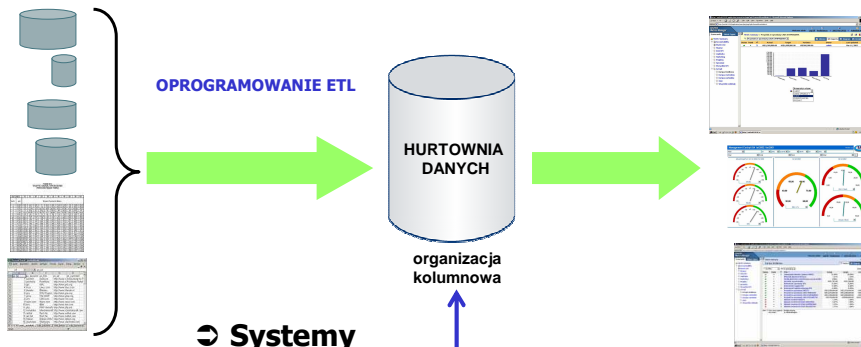
bloki bazy danych

148,00	149,00	148,50	150,00	147,50	148,00	148,00	147,00	148,50
148,50	151,50	150,00	151,50	149,00	152,50	152,00	154,50	164,50
172,50	170,00	169,50	170,50	166,00				

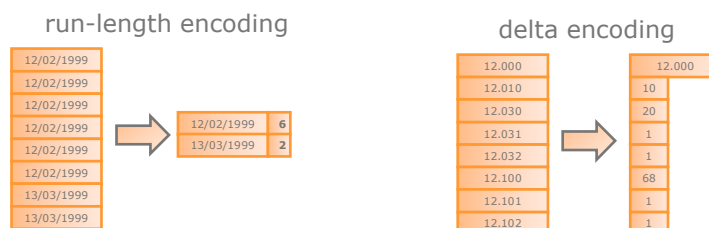
147,00	147,50	146,50	148,00	147,50	142,00	147,50	145,50	147,00
147,00	148,00	149,50	149,00	148,50	146,00	146,00	154,00	153,50
165,00	168,00	163,50	168,00	165,00				



Organizacja kolumnowa (column-oriented)



Optymalizacja (1)





Optimalizacja (2)

discrete domain encoding

Sklep	Rok	Sprzedaz
Alma Stary Browar	01.2006	56 000
Alma Citi Park	01.2006	13 000
Alma Focus Park	01.2006	24 000
Alma Stary Browar	02.2006	52 000
Alma Citi Park	02.2006	18 600
Alma Focus Park	02.2006	21 100
Alma Stary Browar	03.2006	43 200
Alma Citi Park	03.2006	25 700
Alma Focus Park	03.2006	14 700
Alma Stary Browar	04.2006	32 400
Alma Citi Park	04.2006	19 500
Alma Focus Park	04.2006	15 000

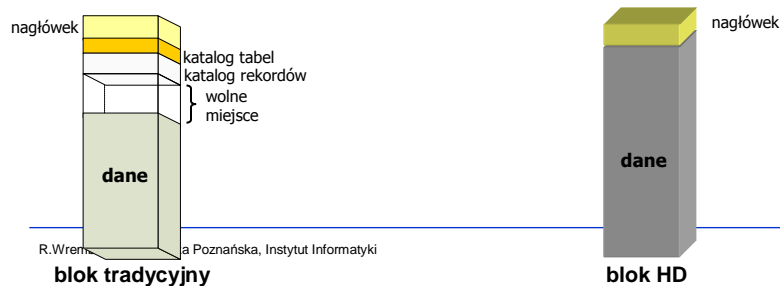
Sklep	Kod
Alma Stary Browar	1
Alma Citi Park	2
Alma Focus Park	3

Sklep	Rok	Sprzedaz
1	01.2006	56 000
2	01.2006	13 000
3	01.2006	24 000
1	02.2006	52 000
2	02.2006	18 600
3	02.2006	21 100
1	03.2006	43 200
2	03.2006	25 700
3	03.2006	14 700
1	04.2006	32 400
2	04.2006	19 500
3	04.2006	15 000



Charakterystyka (1)

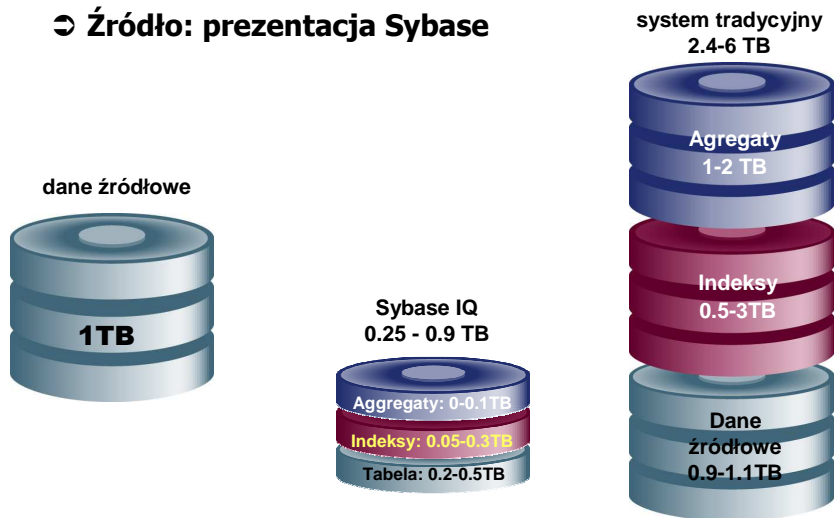
- Efektywne dla operacji projekcji ⇒ mniej danych jest odczytywanych
- Efektywne dla GROUP BY
- Możliwe ułatwienia w implementacji
 - każda kolumna składowana w osobnym pliku zarządzanym przez system operacyjny (np. SADAS)
 - w rozwiązaniu dedykowanym dla HD zasilanie w trybie wsadowym ⇒ uproszczona organizacja bloku danych





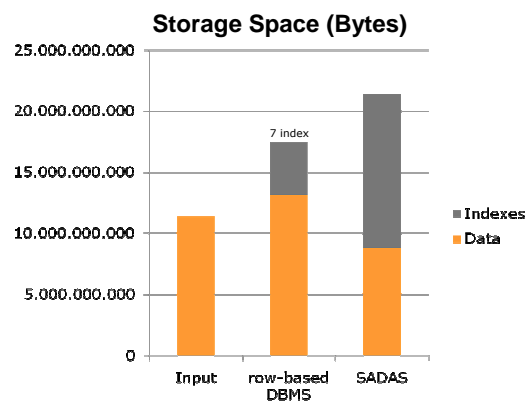
Charakterystyka (2)

➤ Źródło: prezentacja Sybase



Charakterystyka (3)

➤ Źródło: prezentacja AdancedSystems





Inne techniki optymalizacji - SMA

⇒ SMA - Small Materialized Aggregates (G. Moerkotte, VLDB 1998)

- dane na dysku w ramach jednego pliku/przestrzeni tabel są dzielone na obszary (buckets)
- z każdym obszarem jest związany zbiór SMA

BZWBK	2006	Mar	31	148,00	147,00	148,00
BZWBK	2006	Mar	30	149,00	147,50	150,50
BZWBK	2006	Mar	29	148,50	146,50	148,50
BZWBK	2006	Mar	28	150,00	148,00	150,00
BZWBK	2006	Mar	27	147,50	147,50	150,00

SMA obszar1
data min: 27.03.06
data max: 31.03.06
count: 5

BZWBK	2006	Mar	24	148,00	142,00	150,00
BZWBK	2006	Mar	23	148,00	147,50	150,00
BZWBK	2006	Mar	22	147,00	145,50	150,00
BZWBK	2006	Mar	21	148,50	147,00	150,00
BZWBK	2006	Mar	20	148,50	147,00	151,50

SMA obszar2
data min: 20.03.06
data max: 24.03.06
count: 5

BZWBK	2006	Mar	17	151,50	148,00	152,00
BZWBK	2006	Mar	16	150,00	149,50	152,50
BZWBK	2006	Mar	15	151,50	149,00	152,00
BZWBK	2006	Mar	14	149,00	148,50	151,50
BZWBK	2006	Mar	13	152,50	146,00	152,50

SMA obszar3
data min: 13.03.06
data max: 24.03.06
count: 5

- wykorzystanie zapytania z zakresem dat



Inne techniki optymalizacji - ZM

⇒ ZM - Zone Map (Netezza)

- koncepcja podobna do SMA
- dla każdej kolumny i każdego segmentu tworzona jest ZM
- ZM przechowuje wartość MIN i MAX danego atrybutu dla danego segmentu
- zawartość ZM uaktualniana w czasie wczytywania danych do HD
- ZM dla dat: wartości MIN i MAX dla różnych ZM nie będą często się nakładały ⇒ wczytywane dane są skorelowane z czasem, np. sprzedaż



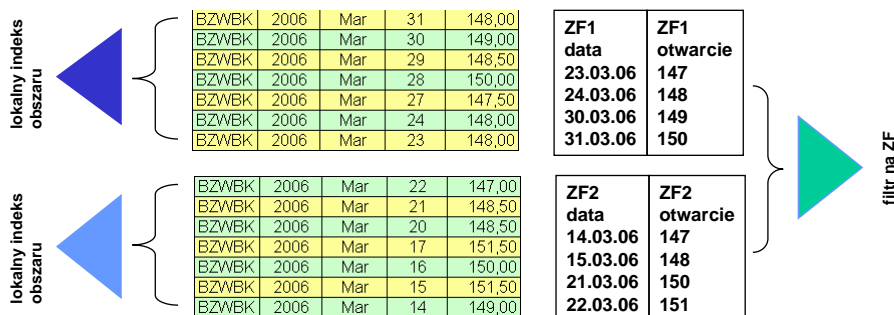
Inne techniki optymalizacji - ZF

☞ ZF - Zone Filter (G. Graefe, DAWAK 2009)

- koncepcja łączy SMA i ZM
- dla każdej kolumny i każdego obszaru tworzony jest ZF
- ZF przechowuje **m** kolejnych wartości MIN i MAX
 - obsługa wartości NULL, jeśli $m=1$, to wartością MIN jest NULL ⇒ nie można wykorzystać do optymalizacji zapytań z zakresem
 - jeśli $m=2$, to wartości najmniejsze są 2: NULL i konkretna wartość
 - $m=3$, $MIN=\{4, 7, 12\}$, zapytanie z warunkiem $=$, np. $a=9$ nie wymaga odczytania tego obszaru
- efektywne jeśli istnieje korelacja pomiędzy wartością atrybutu, a sekwencją wczytywania do HD ⇒ sprzedaż, notowania giełdowe



Inne techniki optymalizacji - ZF



☞ Optymalizacja

- wybór obszarów z wykorzystaniem filtru na ZF
- wyszukanie rekordów w obszarze z wykorzystaniem lokalnego indeksu