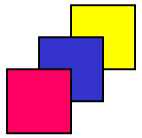
Optymalizacja poleceń SQL

**Optymalizacja kosztowa i regułowa,
dyrektywa AUTOTRACE w SQL*Plus,
statystyki i histogramy, metody dostępu i sortowania,
indeksy typu B* drzewo, indeksy bitmapowe i funkcyjne,
metody połączeń, wskazówki**

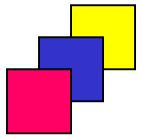


Optymalizacja

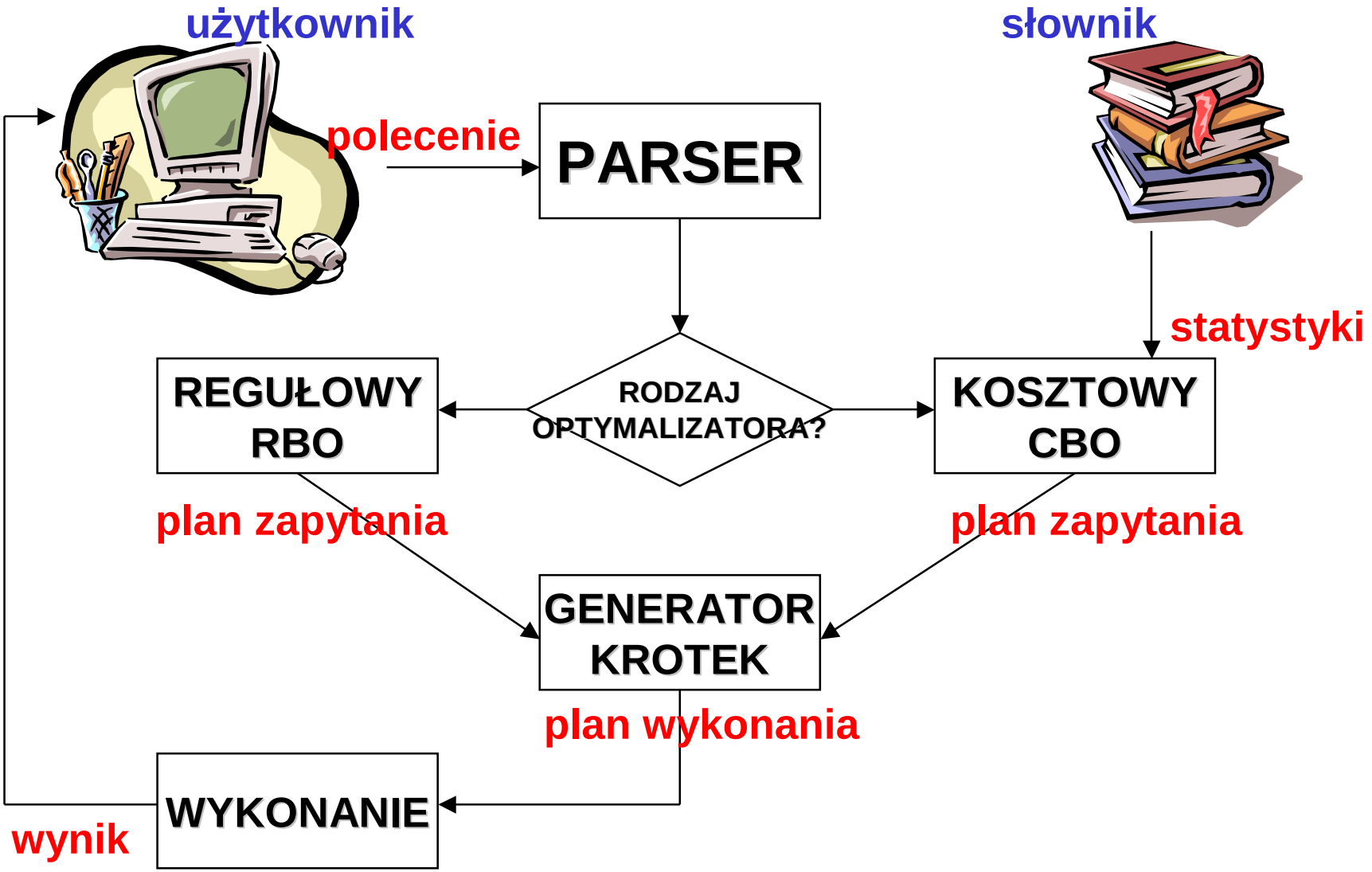
Optymalizacja to proces doboru odpowiednich struktur danych, metod dostępu i operacji (planu wykonania), w celu zminimalizowania kosztu realizacji polecenia.

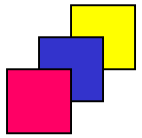
Optymalizacja jest wykonywana przez wyspecjalizowany moduł systemu – optymalizator zapytań.

- **regułowa**
  - **oparta na rankingu metod dostępu do struktur danych**
  - **preferowana dla aplikacji spadkowych**
- **kosztowa**
  - **oparta na szacowaniu kosztu (czas zajętości procesora, liczby operacji we/wy, zajętość pamięci operacyjnej itp.), wykonania wszystkich potencjalnych planów wykonania**
  - **zalecana dla wszystkich nowopowstających aplikacji**
  - **zakłada duże obciążenie systemu: dużą współbieżność operacji, niski współczynnik trafień w bufor danych**



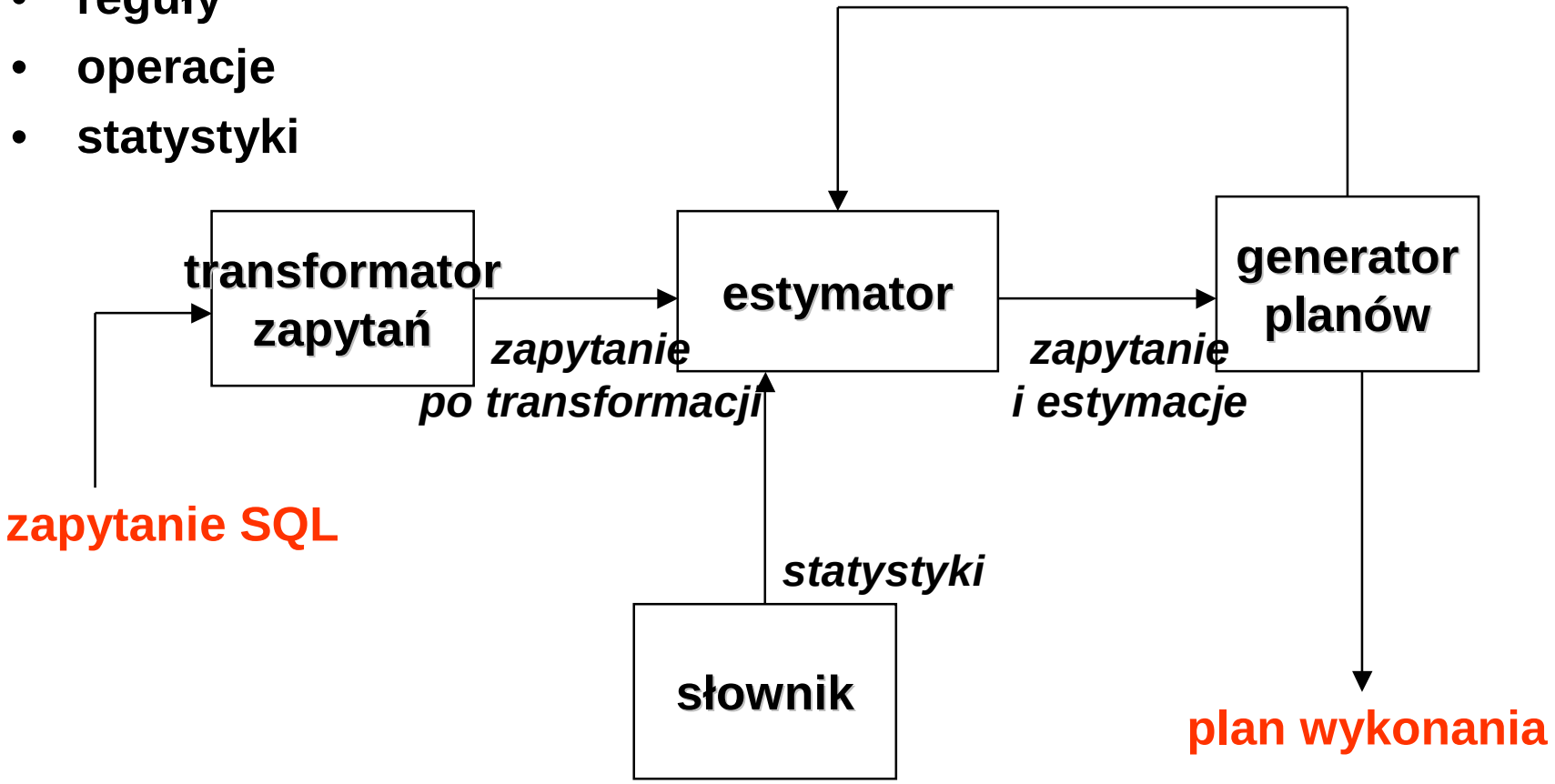
Przetwarzanie polecenia SQL





Optymalizacja polecenia SQL

- struktury danych
- metody dostępu
- reguły
- operacje
- statystyki



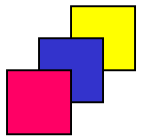
Dyrektywa AUTOTRACE w SQL*Plus (1/2)

```
SQL> @$ORACLE_HOME\SQLPLUS\ADMIN\PLUSTRCE.SQL
SQL> GRANT PLUSTRACE  TO SCOTT
```

```
SQL> SET AUTOTRACE [ ON | OFF ] [ TRACEONLY ]
       [ EXPLAIN ] [ STATISTICS ]
```

```
SQL> SET AUTOTRACE TRACEONLY EXPLAIN
SQL> SELECT nazwisko FROM pracownicy WHERE id_prac=100;

Plan wykonywania
----------------------------------------------------------
   0      SELECT STATEMENT Optimizer= ALL_ROWS (Cost=1 Card=1 Bytes=22)
   1    0   TABLE ACCESS (BY ROWID) OF 'PRACOWNICY' (Cost=1 Card=1 Bytes=22)
   2    1     INDEX (UNIQUE SCAN) OF 'PK_PRAC' (UNIQUE)
SQL> SET AUTOTRACE OFF
```



Dyrektywa AUTOTRACE w SQL*Plus (2/2)

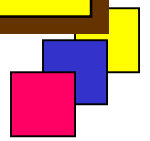
```
SQL> SET AUTOTRACE ON
SQL> SELECT nazwisko FROM pracownicy WHERE placa_pod=480;

ID_PRAC NAZWISKO  ETAT       ID_SZEFA ZATRUDNI  PLACA_POD  PLACA_DOD  ID_ZESP
------- --------- ---------- -------- -------- ---------- ---------- --------
    220 KONOPKA   ASYSTENT         110 93/10/01        480                 20
    230 HAPKE     ASYSTENT         120 92/09/01        480         90      30

Plan wykonywania
----------------------------------------------------------
 0       SELECT STATEMENT Optimizer=ALL_ROWS (Cost=3 Card=4 Bytes=360 )
 1  0    TABLE ACCESS (FULL) OF 'PRACOWNICY' (TABLE) (Cost=3 Card=4 Bytes=360)

Statystyki
----------------------------------------------------------
0 recursive calls
0 db block gets
8 consistent gets
0 physical reads
0 redo size
803 bytes sent via SQL*Net to client
427 bytes received via SQL*Net from client
2 SQL*Net roundtrips to/from client
0 sorts (memory)
0 sorts (disk)
2 rows processed
```

Logiczne I/O

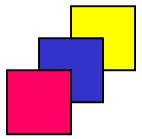


Zmiana celu optymalizacji

Dla bieżącej sesji

- **parametr OPTIMIZER_MODE**
  - **CHOOSE: kosztowa (jeśli są statystyki) lub regułowa (domyślne w Oracle 9i)**
  - **RULE: optymalizacja regułowa**
  - **ALL_ROWS: kosztowa maksymalizująca przepustowość (domyślne w Oracle 10g)**
  - **FIRST_ROWS: kosztowa minimalizująca czas odpowiedzi**
  - **FIRST_ROWS_N: kosztowa minimalizująca łączny czas odczytania pierwszych N (1,10,100,1000) krotek**

```
ALTER SESSION SET OPTIMIZER_MODE = FIRST_ROWS;
```

Statystyki

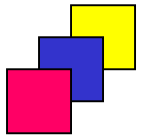
Informacje charakteryzujące struktury danych

dane generowane dla tabeli:

- **liczba wierszy,**
- **liczba bloków danych zawierających dane,**
- **liczba nigdy nie użytych, zaalokowanych bloków danych,**
- **średnia wielkość wolnego miejsca w zajętych blokach danych,**
- **liczba łańcuchowanych wierszy,**
- **średnia wielkość wiersza,**
- **dla wszystkich kolumn liczbę unikalnych wartości oraz wartość minimalną i maksymalną**

dane generowane dla indeksu:

- **wysokość drzewa,**
- **liczbę bloków-liści drzewa,**
- **liczbę unikalnych wartości indeksu,**
- **średnią liczbę bloków-liści przypadającą na jedną wartość klucza indeksu,**
- **średnią liczbę bloków danych (w tabeli) przypadającą jedną wartość klucza indeksu,**
- **współczynnik zgrupowania, który określa na ile wiersze w tabeli są uporządkowane wg klucza indeksy.**

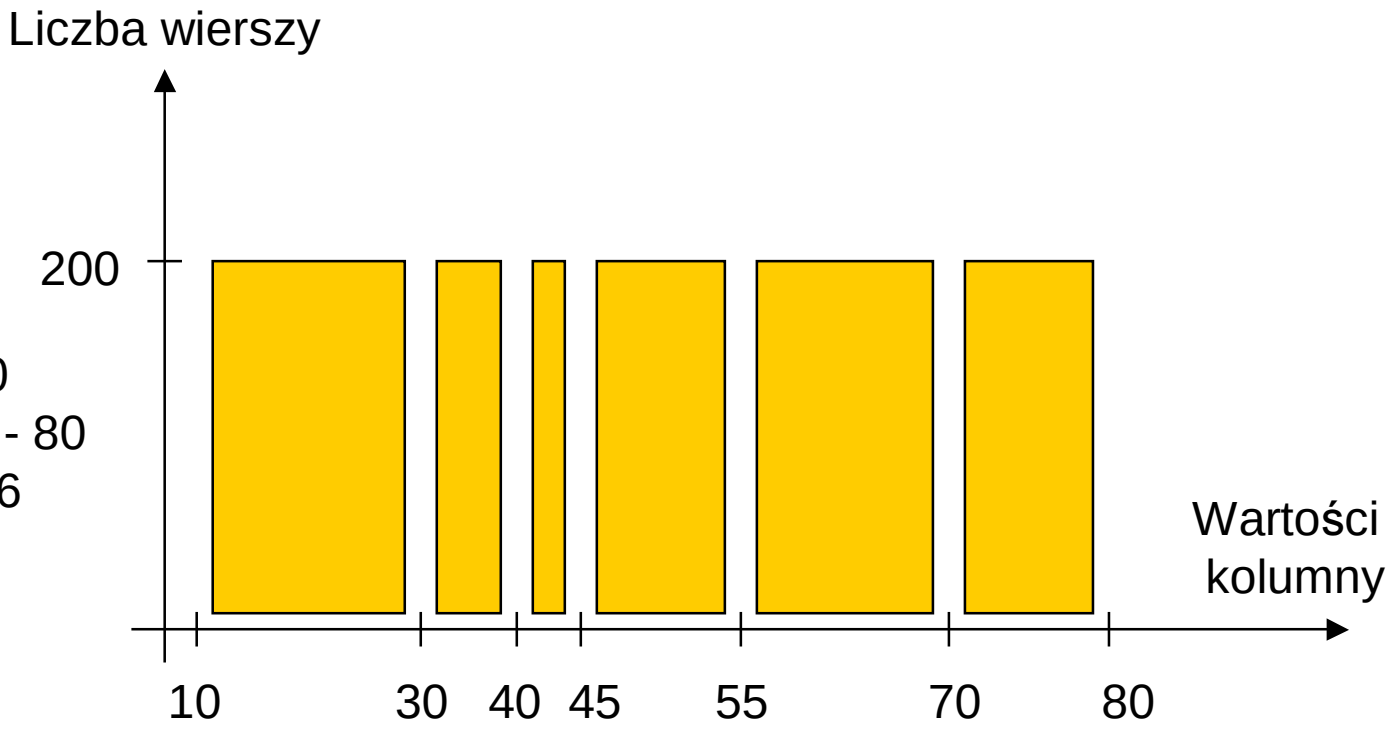


Histogramy

Szczegółowe statystyki opisujące rozkład wartości poszczególnych kolumn, przydatne w szczególności dla optymalizacji wykorzystania indeksów

Przykład:

Dane:
- liczba wierszy 1200
- zakres wartości 10 - 80
- liczba przedziałów 6



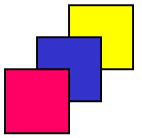
Zbieranie statystyk za pomocą pakietu DBMS_STATS

GATHER_ INDEX_STATS - indeksu
GATHER_TABLE_STATS - tabela, indeks, kolumny
GATHER_SCHEMA_STATS - wszystkie obiekty w schemacie
GATHER_DATABASE_STATS- wszystkie obiekty w bazie danych

```
EXEC DBMS_STATS.GATHER_TABLE_STATS('SCOTT','ZESPOLY');
```

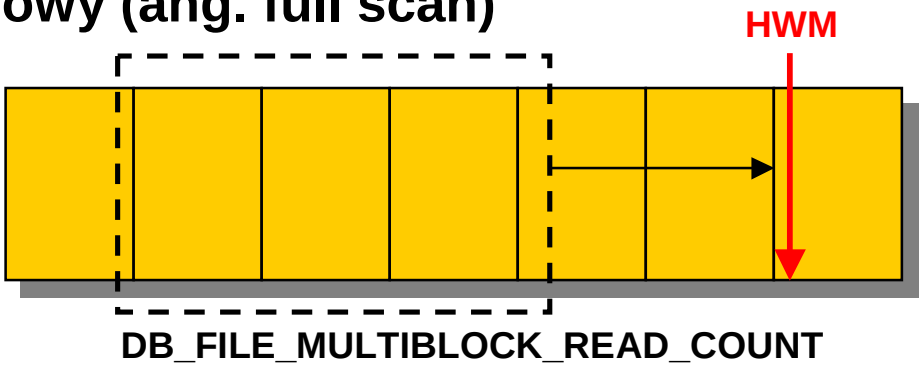
```
EXEC DBMS_STATS.GATHER_TABLE_STATS(
        ownname=>'SCOTT',
        tabname=>'PRACOWNICY',
        estimate_percent=>DBMS_STATS.AUTO_SAMPLE_SIZE,
        method_opt=>'FOR ALL COLUMNS‘
        cascade=>TRUE);
```

```
EXEC DBMS_STATS.DELETE_TABLE_STATS('SCOTT','ZESPOLY');
```

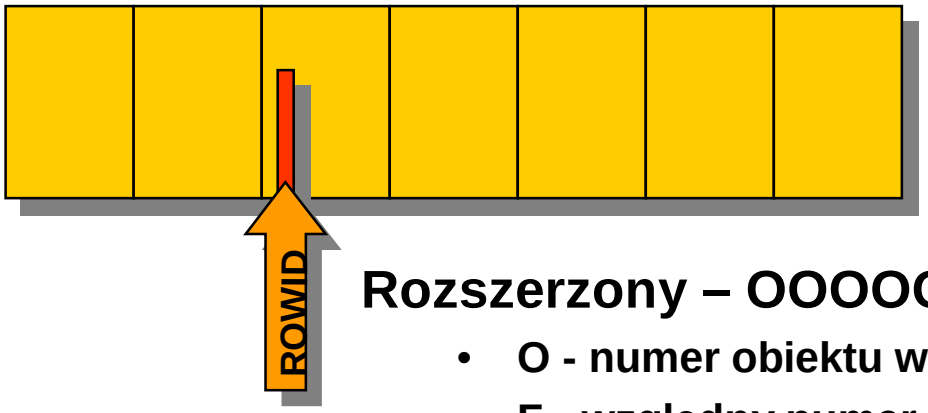


Metody dostępu do tabeli

Przegląd liniowy (ang. full scan)



Dostęp za pomocą adresu rekordu (ang. ROWID)



Rozszerzony – OOOOOO.FFF.BBBBBB.RRR

- O - numer obiektu w bazie danych
- F - względny numer pliku w przestrzeni tabel
- B - numer bloku w pliku
- R - numer rekordu w bloku

Podstawowy – FFFF.BBBBBBBB.RRRR

Rodzaje sortowania

- **ORDER BY: sortowanie wyników zapytania**

```
SELECT * FROM zespoly
ORDER BY adres DESC;
```

- **AGGREGATE: wyliczanie wartości funkcji grupowej**

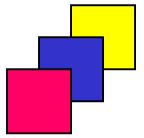
```
SELECT MAX(zatrudniony)
 FROM pracownicy;
```

- **GROUP BY: podział relacji na grupy**

```
SELECT etat, AVG(placa_pod)
 FROM pracownicy GROUP BY etat;
```

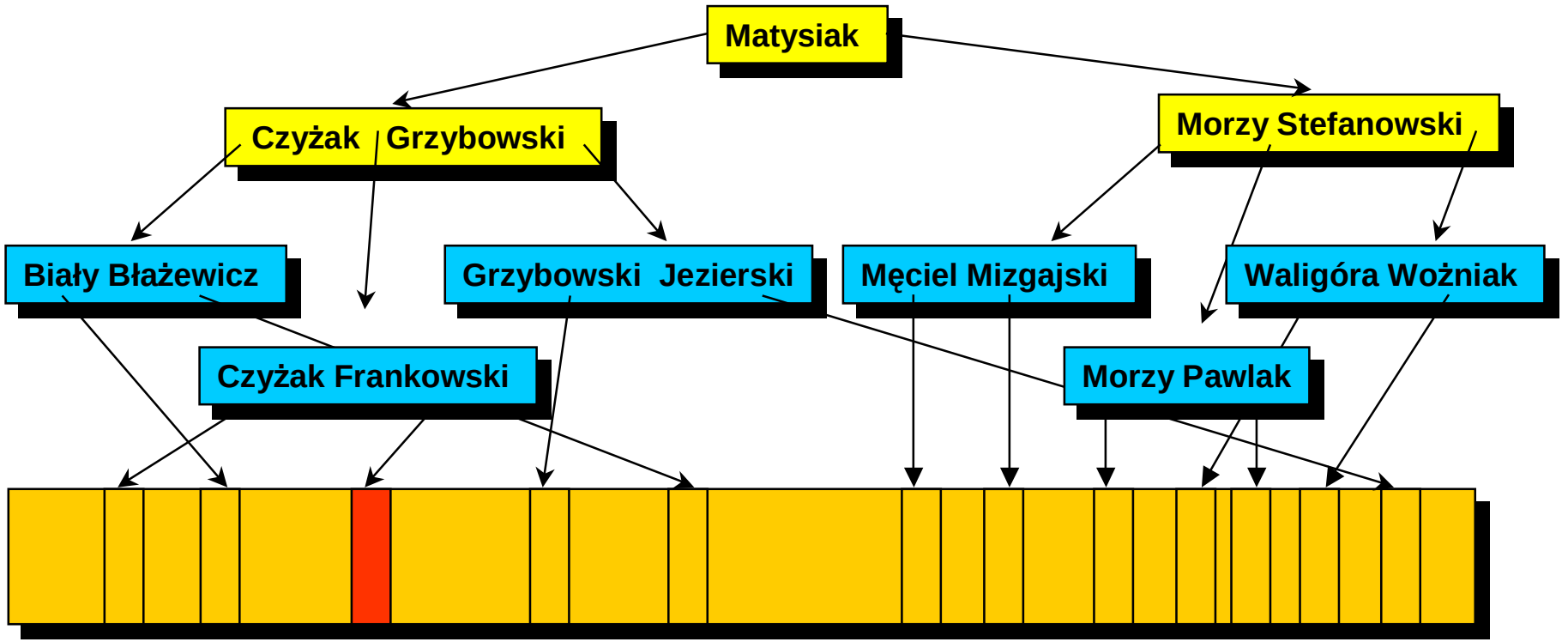
- **UNIQUE: eliminacja duplikatów**

```
SELECT DISTINCT etat
 FROM pracownicy;
```

Indeks B*-drzewo

```
SELECT ETAT, PLACA_POD
FROM PRACOWNICY
WHERE NAZWISKO = 'Frankowski';
```



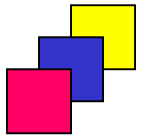
Przesłanki do utworzenia indeksu B*-drzewo

- na atrybutach często wykorzystywanych w warunkach selekcji,
- na atrybutach połączeniowych,
- tylko na atrybutach o dużej selektywności,
- na atrybutach rzadko modyfikowanych,
- na atrybutach będących kluczami obcymi (uniknięcie niepotrzebnego blokowania tabeli podrzędnej w przypadku operacji modyfikacji rekordów nadrzędnych)
- w systemach przetwarzania transakcyjnego - OLTP

```
CREATE [ UNIQUE ] INDEX nazwa ON tabela(atrybut1, atrybut2, ..);
```

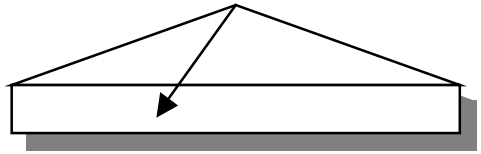
```
CREATE UNIQUE INDEX i_id_prac ON pracownicy(id_prac);
```

```
CREATE INDEX i_complex ON pracownicy(nazwisko, imie);
```

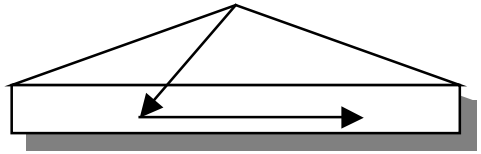


Metody dostępu do indeksu B*- drzewo

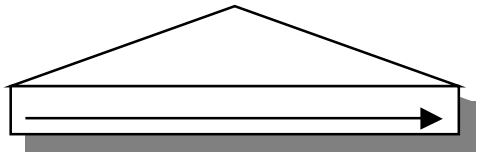
unikalne przeglądnięcie (ang. unique scan)



przeglądnięcie zakresu (ang. range scan)

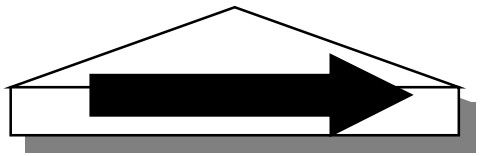


pełne przeglądnięcie (ang. full scan)



- odczyt blok po bloku - nawigacja po liściach
- stosowany również do sortowania

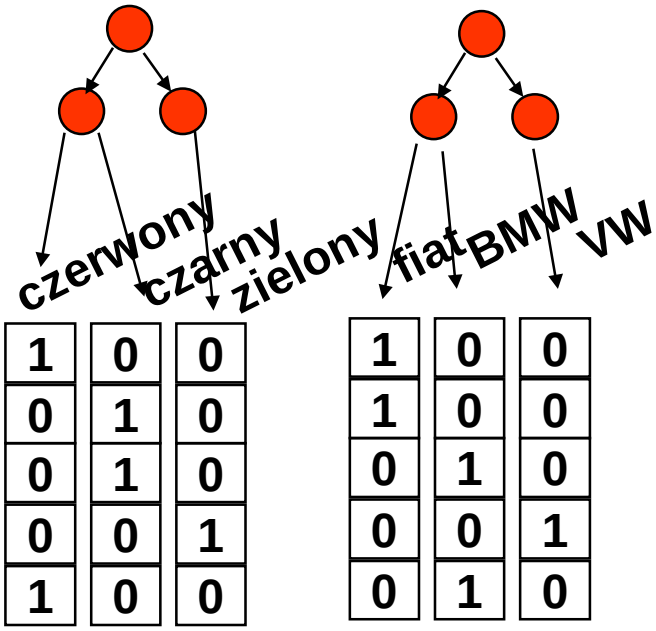
szybkie pełne przeglądniecie (ang. fast full scan)



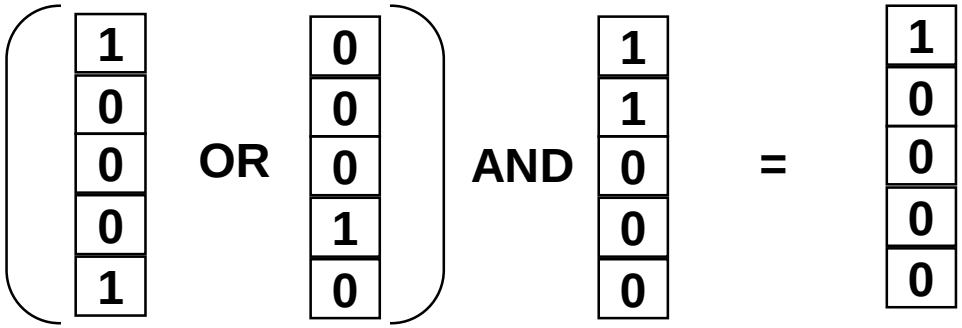
- odczyt wieloblokowy
- stosowany zamiast full table scan

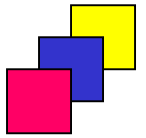
Indeks bitmapowy

PWG01425	czerwony	fiat
WAW3456	czarny	fiat
POZ3756	czarny	BMW
KTW3756	zielony	VW
PNR8956	czerwony	BMW




```
SELECT count(*) FROM samochody
WHERE kolor IN
      ('czerwony', ‘zielony')
      AND marka='fiat'
```



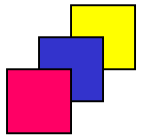


Przesłanki do utworzenia indeksu bitmapowego

- w systemach przetwarzania analitycznego - OLAP,
- na atrybutach o małej selektywności,
- na atrybutach rzadko modyfikowanych
- dla zapytań z poszukiwaniem wartości pustych
- dla zapytań z dużą liczbą warunków OR i AND

```
CREATE BITMAP INDEX nazwa ON tabela (atrybut);
```

```
CREATE BITMAP INDEX b_etat ON pracownicy(etat);
```



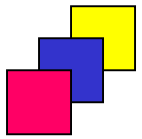
Indeks B*-drzewo vs. bitmapowy

B* - drzewo

- skuteczny dla atrybutów z dużą dziedziną wartości
- efektywne wykonywanie operacji koniunkcji
- wielkość słabo zależna od wielkości dziedziny atrybutu
- bardzo wysoka współbieżność modyfikacji - blokada pojedynczego klucza indeksu
- niski koszt pojedynczej modyfikacji - modyfikacja pojedynczego klucza indeksu
- stosunkowo wysoki koszt modyfikacji grupy rekordów - każda wartość modyfikowana oddzielnie
- główne zastosowanie OLTP

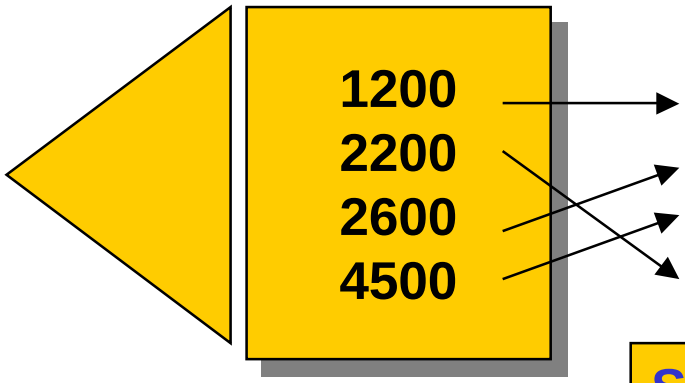
Bitmapowy

- skuteczny dla atrybutów z małą dziedziną wartości
- efektywne wykonywanie operacji alternatywy i koniunkcji
- wielkość bardzo silnie zależna od wielkości dziedziny atrybutu
- niska współbieżność modyfikacji - blokada całej bitmapy
- wysoki koszt pojedynczej modyfikacji - modyfikacji całej bitmapy (kompresja)
- stosunkowo niski koszt modyfikacji grupy rekordów - modyfikacja grupowa z możliwością zrównoleglenia
- główne zastosowanie OLAP



Indeksy oparte na wyrażeniach

```
CREATE INDEX sum_placa ON
pracownicy (placa_pod+nvl(placa_dod,0));
```



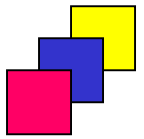
nazwisko	płaca_pod	płaca_dod
Malina	1000	200
Nowak	1400	1200
Kowal	4500	
scott	1700	500

```
SELECT * FROM pracownicy
WHERE placa_pod+nvl(placa_dod,0) < 300;
```

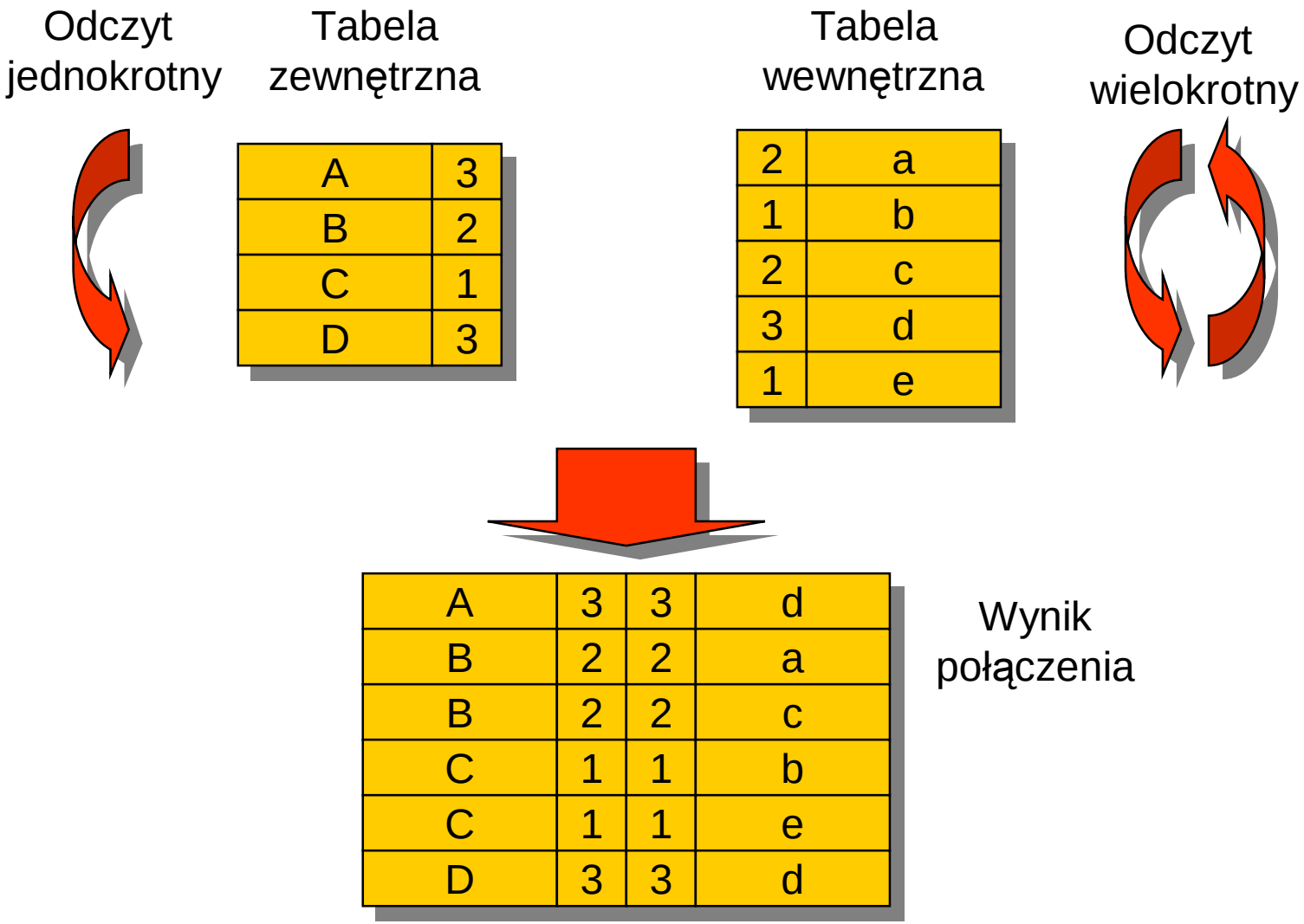
Struktura fizyczna:

- **Bitmapowe**
- **B*-drzewa**

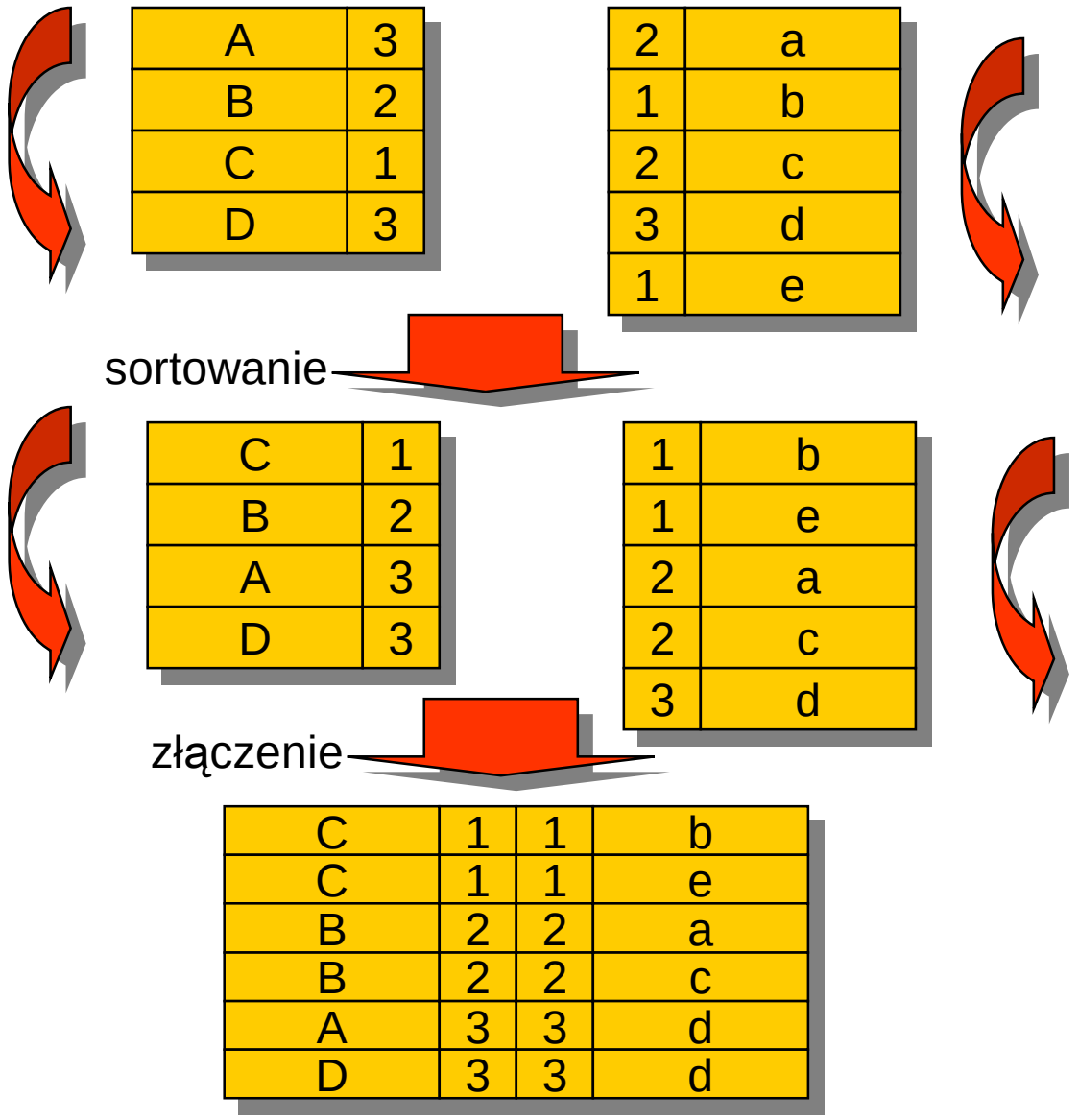
przyśpieszają operacje (selekcja, połączenie) na wyrażeniach,
mogą zawierać funkcje zdefiniowane przez użytkownika
zalecane dla wyrażeń umieszczanych w klauzuli ORDER BY



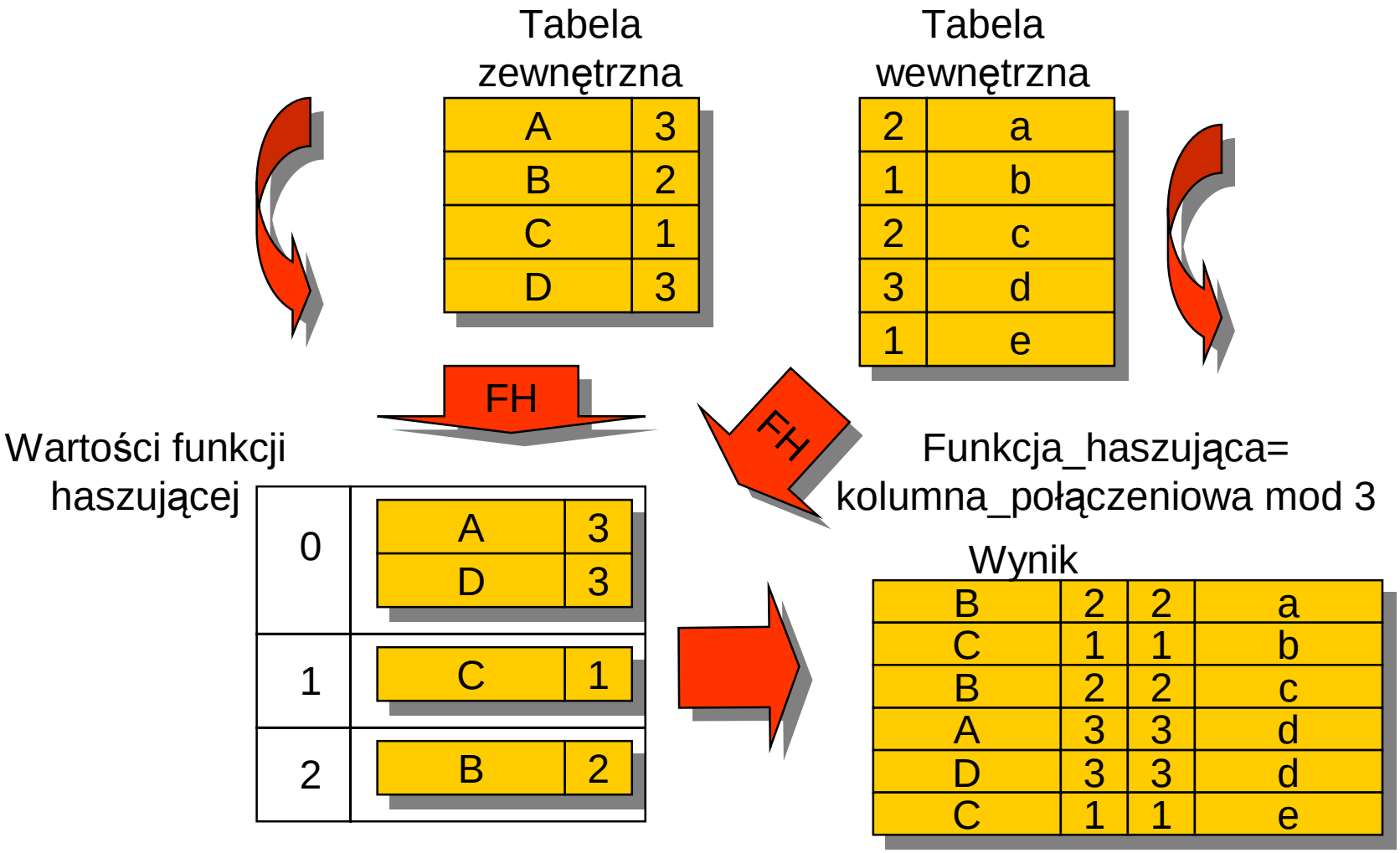
Połączenia - nested loop



Połączenia - sort merge



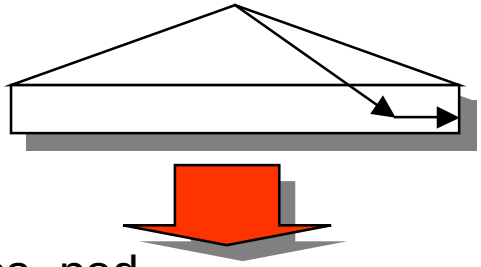
Połączenia - hash join



Połączenia indeksów

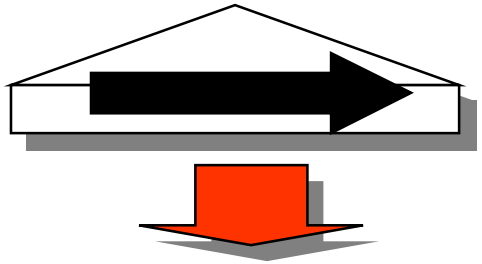
```
SELECT id_prac
FROM pracownicy
WHERE placa_pod >1000;
```

Range scan(indeks na *placa_pod)*



placa_pod	ROWID
1600	00000001.001.001
...	...
2000	00000001.0A1.01E

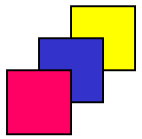
Fast Full Scan(indeksu na *id_prac)*



ROWID	id_prac
00000001.0A1.01E	120
...	...
00000001.001.001	140

join (hash)

120
...
140



Wskazówki (ang. hints)

Wskazówki umożliwiają określenie następujących elementów pracy optymalizatora:

- **rodzaj optymalizatora,**
- **cel optymalizacji,**
- **sposób dostępu do danych,**
- **kolejność łączonych tabel dla operacji połączenia,**
- **sposób realizacji połączenia**

Wskazówki umieszcza się w komentarzu bezpośrednio po klauzulach SELECT, INSERT, UPDATE, DELETE, przy czym pierwszym znakiem wskazówki musi być + (plus)

```
SELECT /*+ FULL_SCAN(PRACOWNICY) */ nazwisko
FROM pracownicy WHERE id_prac=100;
```