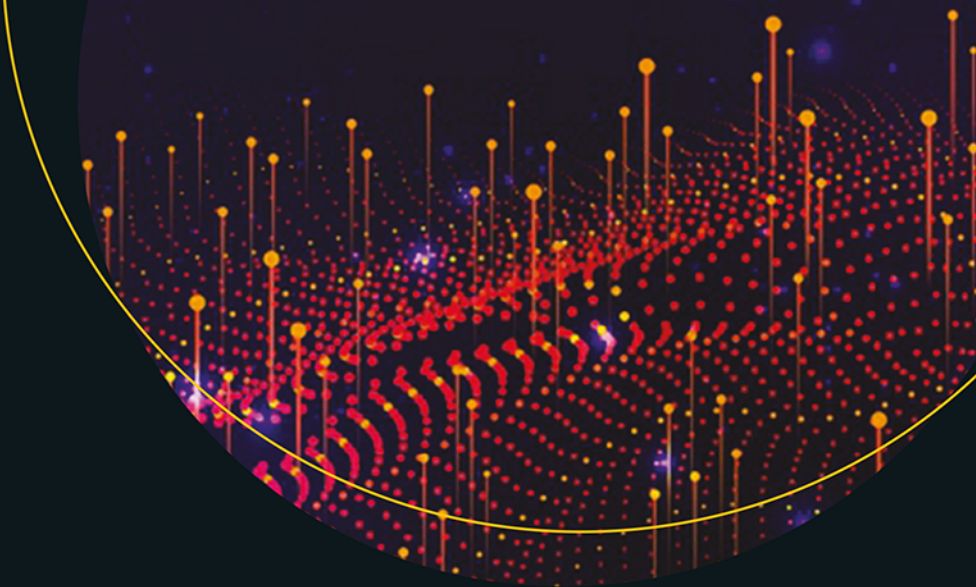




# Praktyczna nauka SQL dla Oracle

Wykorzystaj ogromne możliwości  
bazy danych Oracle

---

Kim Berg Hansen

Tytuł oryginału: Practical Oracle SQL: Mastering the Full Power of Oracle Database

Tłumaczenie: Robert Górczyński

ISBN: 978-83-283-8733-1

First published in English under the title Practical Oracle SQL: Mastering the Full Power of Oracle Database by Kim Berg Hansen, edition: 1

Copyright © 2020 Kim Berg Hansen

This edition has been translated and published under licence from APress Media, LLC, part of Springer Nature. APress Media, LLC, part of Springer Nature takes no responsibility and shall not be made liable for the accuracy of the translation.

Polish edition copyright © 2022 by Helion S.A.  
All rights reserved.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Helion S.A.

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 231 22 19, 32 230 98 63

e-mail: [helion@helion.pl](mailto:helion@helion.pl)

WWW: <https://helion.pl> (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:

<https://ftp.helion.pl/przyklady/prnsql.zip>

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<https://helion.pl/user/opinie/prnsql>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

# Spis treści

|                    |                                                                                     |           |
|--------------------|-------------------------------------------------------------------------------------|-----------|
|                    | <b>O autorze .....</b>                                                              | <b>11</b> |
|                    | <b>Podziękowania .....</b>                                                          | <b>12</b> |
|                    | <b>Wprowadzenie .....</b>                                                           | <b>13</b> |
| <b>Część I</b>     | <b>Podstawy języka SQL .....</b>                                                    | <b>19</b> |
| <b>Rozdział 1.</b> | <b>Korelacja widoków osadzonych .....</b>                                           | <b>21</b> |
|                    | Produkty i ich sprzedaż w przykładowej firmie .....                                 | 21        |
|                    | Podzapytania skalarne i wiele kolumn .....                                          | 23        |
|                    | Skorelowany widok osadzony .....                                                    | 26        |
|                    | Skorelowany widok osadzony i złączenie typu OUTER .....                             | 28        |
|                    | Podsumowanie .....                                                                  | 31        |
| <b>Rozdział 2.</b> | <b>Problemy związane z operacjami na zbiorach .....</b>                             | <b>32</b> |
|                    | Zbiory przedstawiające rodzaje piwa .....                                           | 33        |
|                    | Operatory zbioru .....                                                              | 34        |
|                    | Konkatenacja zbioru .....                                                           | 36        |
|                    | Trzy operatory zbioru .....                                                         | 38        |
|                    | Operatory wielozbioru .....                                                         | 40        |
|                    | Operator MULTiset UNION .....                                                       | 41        |
|                    | Operator MULTiset INTERSECT .....                                                   | 43        |
|                    | Operator MULTiset EXCEPT .....                                                      | 44        |
|                    | Operator MINUS kontra EXCEPT .....                                                  | 45        |
|                    | Podsumowanie .....                                                                  | 48        |
| <b>Rozdział 3.</b> | <b>Dziel i rządź dzięki użyciu faktoringu podzapytania .....</b>                    | <b>49</b> |
|                    | Dane dotyczące produktów i sprzedaży .....                                          | 49        |
|                    | Najlepsze lata pod względem sprzedaży piwa o najmniejszej zawartości alkoholu ..... | 50        |
|                    | Modularyzacja za pomocą klauzuli WITH .....                                         | 53        |
|                    | Wielokrotne używanie tego samego podzapytania .....                                 | 55        |
|                    | Wyświetlanie nazw kolumn .....                                                      | 58        |
|                    | Podsumowanie .....                                                                  | 61        |

|                    |                                                                      |            |
|--------------------|----------------------------------------------------------------------|------------|
| <b>Rozdział 4.</b> | <b>Drzewo obliczeń i rekurencja .....</b>                            | <b>62</b>  |
|                    | Butelki w kartonach na palecie .....                                 | 62         |
|                    | Mnożenie ilości hierarchicznych .....                                | 64         |
|                    | Rekurencyjna faktoryzacja podzapytania .....                         | 66         |
|                    | Dynamiczny SQL w funkcji PL/SQL .....                                | 71         |
|                    | Podsumowanie .....                                                   | 73         |
| <b>Rozdział 5.</b> | <b>Funkcje zdefiniowane w języku SQL .....</b>                       | <b>74</b>  |
|                    | Tabela z danymi zawartości alkoholu w piwie .....                    | 74         |
|                    | Stężenie alkoholu we krwi .....                                      | 76         |
|                    | Zdefiniowanie funkcji za pomocą PRAGMA UDF .....                     | 77         |
|                    | Zdefiniowanie funkcji za pomocą klauzuli WITH .....                  | 79         |
|                    | Hermetyzacja kodu w widoku .....                                     | 82         |
|                    | Podsumowanie .....                                                   | 83         |
| <b>Rozdział 6.</b> | <b>Obliczenia iteracyjne z użyciem danych wielowymiarowych .....</b> | <b>84</b>  |
|                    | „Gra w życie” Johna Conwaya .....                                    | 84         |
|                    | Liczba żywych sąsiadów obliczona za pomocą klauzuli MODEL .....      | 86         |
|                    | Iteracja przez generacje .....                                       | 91         |
|                    | Podsumowanie .....                                                   | 98         |
| <b>Rozdział 7.</b> | <b>Anulowanie przestawienia kolumn na rekordy .....</b>              | <b>99</b>  |
|                    | Dane otrzymane w kolumnach .....                                     | 99         |
|                    | Anulowanie przestawienia kolumn .....                                | 100        |
|                    | Samodzielne anulowanie przestawienia kolumn .....                    | 102        |
|                    | Więcej niż tylko jeden wymiar i/lub miara .....                      | 104        |
|                    | Używanie tabel wymiarów .....                                        | 109        |
|                    | Dynamiczne mapowanie tabeli wymiaru .....                            | 111        |
|                    | Podsumowanie .....                                                   | 115        |
| <b>Rozdział 8.</b> | <b>Przestawianie rekordów na kolumny .....</b>                       | <b>116</b> |
|                    | Tabele używane podczas przestawiania kolumn .....                    | 117        |
|                    | Przestawianie kolumny pojedynczej miary i pojedynczego wymiaru ..... | 119        |
|                    | Ręczne przeprowadzenie operacji przestawiania kolumn .....           | 121        |
|                    | Wiele miar .....                                                     | 122        |
|                    | Wiele wymiarów .....                                                 | 124        |
|                    | Podsumowanie .....                                                   | 126        |
| <b>Rozdział 9.</b> | <b>Podział ograniczonego tekstu .....</b>                            | <b>127</b> |
|                    | Ulubione piwa użytkowników i pisane przez nich recenzje .....        | 127        |
|                    | Ograniczanie pojedynczych wartości .....                             | 128        |
|                    | Potokowana funkcja tabeli .....                                      | 129        |
|                    | Funkcja tabeli wbudowanego schematu APEX .....                       | 132        |
|                    | Czysty kod SQL z generatorem rekordu .....                           | 133        |
|                    | Traktowanie ciągu tekstowego jako tablicy JSON .....                 | 135        |
|                    | Ograniczone wiele wartości .....                                     | 136        |
|                    | Niestandardowa funkcja tabeli pochodząca z ODCI .....                | 137        |
|                    | Połączenie funkcji apex_string.split() i substr() .....              | 140        |
|                    | Generator rekordów i wywołanie regexp_substr() .....                 | 141        |
|                    | Konwersja na format JSON .....                                       | 142        |
|                    | Podsumowanie .....                                                   | 145        |

|                                                                                                   |                |
|---------------------------------------------------------------------------------------------------|----------------|
| <b>Rozdział 10. Tworzenie ograniczonego tekstu .....</b>                                          | <b>146</b>     |
| Lista produktów w postaci ograniczonego ciągu tekstowego .....                                    | 146            |
| Agregacja ciągu tekstowego .....                                                                  | 148            |
| Agregacja z użyciem funkcji listagg() .....                                                       | 148            |
| Funkcja agregacji collect() .....                                                                 | 150            |
| Niestandardowa funkcja agregacji stragg() .....                                                   | 152            |
| Funkcja agregacji xmlagg() .....                                                                  | 156            |
| Gdy wartość nie mieści się w typie varchar2 .....                                                 | 157            |
| Pobranie jedynie pierwszej części wyniku .....                                                    | 158            |
| Próba zmieszczenia zmniejszonego zbioru danych .....                                              | 159            |
| Używanie typu clob zamiast varchar2 .....                                                         | 160            |
| Podsumowanie .....                                                                                | 162            |
| <br><b>Część II Funkcje analityczne .....</b>                                                     | <br><b>163</b> |
| <b>Rozdział 11. Klauzule partycjonowania oraz definiowania kolejności i okien .....</b>           | <b>165</b>     |
| Suma ilości .....                                                                                 | 165            |
| Składnia analityczna .....                                                                        | 167            |
| Partycje .....                                                                                    | 168            |
| Kolejność i okna .....                                                                            | 169            |
| Elastyczność klauzuli okna .....                                                                  | 172            |
| Definiowanie okna na podstawie wartości zakresu .....                                             | 175            |
| Niebezpieczeństwo związane z oknem domyślnym .....                                                | 176            |
| Podsumowanie .....                                                                                | 179            |
| <br><b>Rozdział 12. Udzielanie odpowiedzi na pytania typu Najlepsze-N .....</b>                   | <b>181</b>     |
| Najlepsze-N rekordów danych o sprzedaży .....                                                     | 181            |
| Który rodzaj Najlepsze-3 masz na myśli? .....                                                     | 182            |
| Dane dotyczące sprzedaży piwa .....                                                               | 183            |
| Tradycyjna metoda rownum .....                                                                    | 186            |
| Funkcje analityczne dotyczące rankingu .....                                                      | 187            |
| Pobieranie tylko pierwszych rekordów .....                                                        | 189            |
| Obsługa remisu .....                                                                              | 190            |
| Na co nie pozwala klauzula ograniczająca? .....                                                   | 192            |
| Najlepsze-N rekordów w wielu partycjach .....                                                     | 194            |
| Sztuczka przeznaczona do zastosowania w klauzuli ograniczającej rekordy .....                     | 196            |
| Podsumowanie .....                                                                                | 197            |
| <br><b>Rozdział 13. Zbiór uporządkowany za pomocą sumy kroczącej .....</b>                        | <b>198</b>     |
| Dane używane podczas pobierania produktów .....                                                   | 199            |
| Tworzenie zapytania SQL pobierającego dane .....                                                  | 200            |
| Rozwiązanie pierwszego problemu za pomocą kolejności FIFO .....                                   | 201            |
| Łatwa zmiana reguł dotyczących pobierania produktów .....                                         | 205            |
| Rozwiązanie problemu optymalnej trasy pobierania produktów .....                                  | 207            |
| Rozwiązanie problemu pobierania produktów partiami .....                                          | 210            |
| Dokończenie pracy nad kodem SQL pomagającym<br>w określeniu kolejności pobierania produktów ..... | 217            |
| Podsumowanie .....                                                                                | 218            |

|                                                                              |            |
|------------------------------------------------------------------------------|------------|
| <b>Rozdział 14. Analizowanie dzienników zdarzeń za pomocą funkcji lead()</b> | <b>220</b> |
| Dziennik zdarzeń pobierania produktów                                        | 221        |
| Analiza przyjazdów i odjazdów                                                | 223        |
| Analizowanie czynności pobierania produktów                                  | 226        |
| Ukończenie analizy cykli pobierania produktów                                | 229        |
| Zapowiedź — dopasowanie wzorca rekordu                                       | 232        |
| Podsumowanie                                                                 | 234        |
| <b>Rozdział 15. Prognozowanie z użyciem regresji liniowej</b>                | <b>235</b> |
| Prognozowanie sprzedaży                                                      | 236        |
| Szeregi czasowe                                                              | 237        |
| Obliczanie punktu wyjścia dla regresji                                       | 239        |
| Regresja liniowa                                                             | 242        |
| Ostateczna prognoza                                                          | 246        |
| Podsumowanie                                                                 | 248        |
| <b>Rozdział 16. Suma krocząca podczas prognozowania osiągnięcia minimum</b>  | <b>249</b> |
| Stany magazynowe, budżet i zamówienia                                        | 249        |
| Dane                                                                         | 251        |
| Akumulacja aż do osiągnięcia zera                                            | 253        |
| Uzupełnianie stanów po osiągnięciu minimum                                   | 255        |
| Podsumowanie                                                                 | 260        |
| <b>Część III Dopasowanie wzorca rekordu</b>                                  | <b>261</b> |
| <b>Rozdział 17. Wzorce w górę i w dół</b>                                    | <b>263</b> |
| Przykład wykorzystujący dane giełdowe                                        | 263        |
| Klasyfikacja wzrostów i spadków                                              | 264        |
| Spadki i wzrosty prowadzą do wygenerowania kształtu V                        | 269        |
| Sprawdzenie, czy klasyfikacja SAME wciąż jest potrzebna                      | 273        |
| $V + V =$ kształt W                                                          | 276        |
| Nakładające się kształty W                                                   | 279        |
| Podsumowanie                                                                 | 281        |
| <b>Rozdział 18. Grupowanie danych za pomocą wzorców</b>                      | <b>283</b> |
| Grupowanie dwóch zbiorów danych                                              | 283        |
| Trzy warunki grupowania                                                      | 284        |
| Grupowanie kolejnych danych                                                  | 284        |
| Grupowanie do chwili, gdy przerwa stanie się zbyt duża                       | 293        |
| Grupowanie aż do osiągnięcia ustalonej granicy                               | 295        |
| Podsumowanie                                                                 | 297        |
| <b>Rozdział 19. Łączenie zakresów dat</b>                                    | <b>298</b> |
| Okresy zatrudnienia                                                          | 298        |
| Ważność czasowa                                                              | 302        |
| Złączanie nakładających się okresów                                          | 303        |
| Próba porównania z poprzednim rekordem                                       | 304        |
| Lepsze porównanie z maksymalną datą końcową                                  | 306        |
| Obsługa dat null                                                             | 310        |
| Podsumowanie                                                                 | 311        |

|                                                                             |            |
|-----------------------------------------------------------------------------|------------|
| <b>Rozdział 20. Wyszukiwanie nagłych skoków .....</b>                       | <b>312</b> |
| Historia licznika odwiedzin strony internetowej .....                       | 313        |
| Dane licznika .....                                                         | 314        |
| Wzorce w niezmodyfikowanych danych licznika odwiedzin .....                 | 315        |
| Dzienna liczba odwiedzin strony .....                                       | 319        |
| Wzorce w danych dotyczących dziennych odwiedzin strony .....                | 320        |
| Znacznie bardziej skomplikowane wzorce .....                                | 326        |
| Podsumowanie .....                                                          | 328        |
| <b>Rozdział 21. Optymalizacja pakowania .....</b>                           | <b>329</b> |
| Produkty, które mają być spakowane w kartony .....                          | 329        |
| Optymalizacja pakowania za pomocą nieograniczonej liczby kartonów           |            |
| o ograniczonej pojemności .....                                             | 331        |
| Optymalizacja pakowania do mniejszych kartonów .....                        | 338        |
| Optymalizacja pakowania za pomocą ograniczonej liczby kartonów              |            |
| o nieograniczonej pojemności .....                                          | 340        |
| Podsumowanie .....                                                          | 346        |
| <b>Rozdział 22. Zliczanie elementów potomnych w strukturze drzewa .....</b> | <b>347</b> |
| Hierarchiczne drzewo pracowników .....                                      | 347        |
| Zliczanie podwładnych na wszystkich poziomach .....                         | 349        |
| Zliczanie rekordów za pomocą dopasowania wzorca .....                       | 350        |
| Szczegóły każdego dopasowania .....                                         | 352        |
| Eksperymentowanie z danymi wyjściowymi .....                                | 357        |
| Podsumowanie .....                                                          | 359        |
| <b>Skorowidz .....</b>                                                      | <b>361</b> |

## ROZDZIAŁ 5.



# Funkcje zdefiniowane w języku SQL

Jedną z pięknych stron języka SQL w Oracle jest możliwość jego łatwej rozbudowy przez tworzenie funkcji wywoływanych w kodzie SQL. Zwykle to będą funkcje PL/SQL, choć w niektórych przypadkach mogą być utworzone również w językach C lub Java. Dzięki nowemu silnikowi w przyszłości będzie możliwe tworzenie procedur składowanych i funkcji w wielu różnych językach.

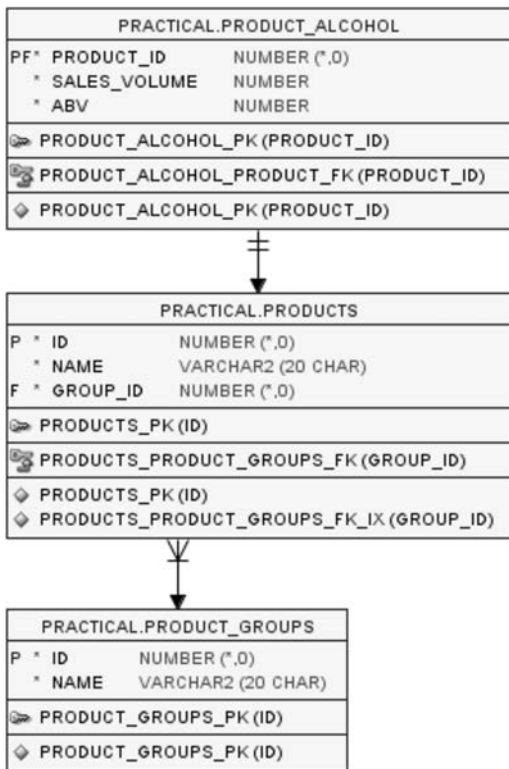
Trzeba w tym miejscu wyraźnie podkreślić, że kod SQL i PL/SQL jest wykonywany przez dwa oddzielne silniki, między którymi istnieją niewielkie różnice w zakresie np. sposobu obsługi zmiennych, typów danych i pamięci. W trakcie każdej operacji wywołania przez SQL funkcji PL/SQL lub na odwrót, gdy PL/SQL wykonuje statyczny bądź dynamiczny kod SQL, dane są przekazywane między wspomnianymi silnikami. Podczas tej operacji może się odbywać również konwersja — to nazywamy przełączaniem kontekstu.

Operacja przełączania kontekstu jest bardzo krótka i zwykle nie ma powodów do zastanawiania się nad nią. Jeżeli jednak funkcja będzie wywoływana z poziomu kodu SQL tysiące razy w ciągu sekundy, wówczas czasy poszczególnych operacji przełączania kontekstu zsumują się i ostatecznie będą miały pokaźny udział w całkowitym czasie wykonania zapytania. W wersji 12.1 serwera Oracle istnieje możliwość minimalizacji liczby operacji przełączania kontekstu, a tym samym staje się ono ledwo zauważalne.

## Tabela z danymi zawartości alkoholu w piwie

Aby pokazać minimalną funkcję przełączania kontekstu w kodzie SQL, posłużę się tabelą `product_alcohol` o strukturze pokazanej na rysunku 5.1.

W tej tabeli dla każdego rodzaju piwa jest przechowywana objętość (wyrażona w mililitrach) jednostki sprzedaży (butelka lub inny pojemnik) oraz procentowa zawartość alkoholu. Kod przedstawiony na listingu 5.1 powoduje wyświetlenie danych dla piwa w grupie 142, którą jest Stout (to względnie mocne i ciemne piwo).



**Rysunek 5.1.** Tabela `product_alcohol` zawiera dane niezbędne do obliczania zawartości alkoholu w piwie

**Listing 5.1.** Dotyczące alkoholu dane dla piwa w grupie Stout

```

SQL> select
2     p.id as p_id
3     , p.name
4     , pa.sales_volume as vol
5     , pa.abv
6 from products p
7 join product_alcohol pa
8     on pa.product_id = p.id
9 where p.group_id = 142
10 order by p.id;
```

Piwo Reindeer Fuel jest rozlewane w butelkach półlitrowych (500 ml), a jego procentowa zawartość alkoholu wynosi tylko 6%. Dwa pozostałe piwa w tej grupie są rozlewane w standardowych butelkach 330 ml, a ich procentowa zawartość alkoholu jest większa.

| P_ID | NAME             | VOL | ABV |
|------|------------------|-----|-----|
| 4040 | Coalminers Sweat | 330 | 8,5 |
| 4160 | Reindeer Fuel    | 500 | 6   |
| 4280 | Hoppy Crude Oil  | 330 | 7   |

Te dane można wykorzystać do ustalenia, ile czystego alkoholu zawiera butelka danego piwa. Dzięki temu można obliczyć stężenie alkoholu we krwi po wypiciu danego piwa.

## Stężenie alkoholu we krwi

Firma Good Beer Trading Co musi stosować się do przepisów, zgodnie z którymi na każdym opakowaniu napoju trzeba podać maksymalne stężenie alkoholu we krwi po wypiciu danej butelki piwa. Ta wartość będzie różna dla kobiet i mężczyzn. Zależy od masy ciała — musi być podana dla kobiety o wadze 60 kg i mężczyzny o wadze 80 kg.

Stężenie alkoholu we krwi musi być obliczone jako liczba gramów alkoholu w mililitrze krwi; zwykle jest podawane w procentach. Dlatego też stężenie 0,04 oznacza, że 0,04% krwi to czysty alkohol. Do obliczenia tej wartości można skorzystać ze wzoru Widmarka.

---

■ **Wzór Widmarka** Wyrażona w mililitrach ilość wypitego napoju \* procentowa zawartość alkoholu / 100 = mililitry alkoholu. Mililitry alkoholu \* 0,789 (ciężar właściwy alkoholu) = gramy alkoholu. Waga ciała \* 1000 \* procentowa zawartość wody w ciele danej płci = mililitry płynu w ciele. (W przypadku kobiet procentowa zawartość wody wynosi 55%, a w przypadku mężczyzn to 68%).  
 $100 * \text{gramy alkoholu} / \text{mililitry płynu we krwi} = \text{stężenie alkoholu we krwi}$

---

Po zastosowaniu w kodzie SQL wzoru Widmarka można obliczyć żądane wartości stężenia alkoholu we krwi, jak pokazałem na listingu 5.2.

**Listing 5.2.** Obliczanie stężenia alkoholu we krwi dla kobiet i mężczyzn

```
SQL> select
2     p.id as p_id
3     , p.name
4     , pa.sales_volume as vol
5     , pa.abv
6     , round(
7         100 * (pa.sales_volume * pa.abv / 100 * 0.789)
8         / (80 * 1000 * 0.68)
9     , 3
10    ) bac_m
11    , round(
12        100 * (pa.sales_volume * pa.abv / 100 * 0.789)
13        / (60 * 1000 * 0.55)
14    , 3
15    ) bac_f
16 from products p
17 join product_alcohol pa
18   on pa.product_id = p.id
19 where p.group_id = 142
20 order by p.id;
```

W wierszach od 6. do 10. następuje obliczenie stężenia alkoholu we krwi dla mężczyzny o wadze 80 kg, a w wierszach od 11. do 15. te same obliczenia są przeprowadzane dla kobiety o wadze 60 kg. Mężczyzna ma więcej wody w organizmie (zarówno z powodu płci, jak i większej wagi), więc szybciej spala alkohol i ma mniejsze jego stężenie we krwi.

Te obliczenia dostarczają kolumnom bac\_m i bac\_f wartości, które firma Good Beer Trading Co musi później umieszczać na etykietach butelek piwa i na kartonach.

| P_ID | NAME             | VOL | ABV | BAC_M | BAC_F |
|------|------------------|-----|-----|-------|-------|
| 4040 | Coalminers Sweat | 330 | 8.5 | 0.041 | 0.067 |
| 4160 | Reindeer Fuel    | 500 | 6   | 0.044 | 0.072 |
| 4280 | Hoppy Crude Oil  | 330 | 7   | 0.034 | 0.055 |

Dzięki obliczonym wcześniej wartościom można sprawdzić, czy np. dany kraj to jeden z tych, w których można prowadzić pojazdy mechaniczne przy stężeniu alkoholu we krwi wynoszącym 0,05%. (W niektórych krajach tę wartość podaje się w promilach, wówczas wynosi ona 0,5‰). Wypicie takiego piwa przez kobietę o wadze 60 kg spowoduje przekroczenie przez nią dozwolonego stężenia alkoholu we krwi i narazi ją na odpowiedzialność karną za jazdę po spożyciu. Natomiast w przypadku mężczyzny o wadze 80 kg stężenie alkoholu we krwi będzie poniżej dozwolonej granicy.

---

■ **Uwaga** To są przykładowe dane niezbędne do zilustrowania wzoru zapisanego w kodzie SQL. Rzeczywiste stężenie alkoholu we krwi będzie zależało od wielu innych czynników, m.in. budowy ciała i metabolizmu, więc nie należy go używać do ustalenia, czy można zasiąść za kierownicą po wypiciu kilku butelek piwa. Te przykłady są przeznaczone jedynie do nauki języka SQL — nie biorę odpowiedzialności za mandaty, które możesz otrzymać, a także apeluję o rozsądne spożywanie napojów alkoholowych.

---

Jako programista, prawdopodobnie wyraźnie dostrzegasz, że kod przedstawiający wzór Widmarka powinien zostać umieszczony w funkcji, a nie powielony z nieco innymi wartościami w zapytaniu.

## Zdefiniowanie funkcji za pomocą PRAGMA UDF

W kodzie zamieszczonym na listingu 5.3 najpierw została utworzona zwykła (no prawie zwykła) funkcja PL/SQL wykorzystująca wzór Widmarka do obliczenia stężenia alkoholu we krwi. Wprawdzie to nie ma tutaj żadnego znaczenia, ale zastosowałem się do najlepszej praktyki, polegającej na umieszczaniu funkcji w pakiecie zamiast definiowania jej jako samodzielnej funkcji. Dlatego też postanowiłem utworzyć pakiet formuła przeznaczony dla tego rodzaju funkcji.

**Listing 5.3.** Tworzenie pakietu wzoru dla funkcji obliczającej stężenie alkoholu we krwi

```
SQL> create or replace package formulas
2  is
3      function bac (
4          p_volume in number
5          , p_abv   in number
6          , p_weight in number
7          , p_gender in varchar2
8          ) return number deterministic;
9  end formulas;
10 /
```

Package FORMULAS compiled

```

SQL> create or replace package body formulas
2  is
3      function bac (
4          p_volume in number
5          , p_abv   in number
6          , p_weight in number
7          , p_gender in varchar2
8      ) return number deterministic
9  is
10     PRAGMA UDF;
11  begin
12     return round(
13         100 * (p_volume * p_abv / 100 * 0.789)
14         / (p_weight * 1000 * case p_gender
15                                     when 'M' then 0.68
16                                     when 'F' then 0.55
17                                 end)
18         , 3
19     );
20  end bac;
21 end formulas;
22 /

```

Package Body FORMULAS compiled

Przeznaczenie tego kodu jest dość oczywiste i niejasny może być jedynie wiersz 10. **PRAGMA UDF** (ang. *user-defined function*, czyli funkcja zdefiniowana przez użytkownika) to dyrektywa kompilatora wprowadzona w wersji 12.1. Jej celem jest wskazanie kompilatorowi, że dana funkcja ma być *przede wszystkim* wywoływana z poziomu kodu SQL, a nie PL/SQL.

Jeżeli funkcja zostanie zdefiniowana *bez* tej dyrektywy, zostanie skompilowana w standardowy sposób, co oznacza przełączenie kontekstu podczas wywoływania danej funkcji. Natomiast zdefiniowanie funkcji *razem* z podaną dyrektywą powoduje je skompilowanie w inny sposób, który może zmniejszyć obciążenie związane z przełączaniem kontekstu. Poziom (ewentualnego) zmniejszenia obciążenia może być niezależny od programisty. Do tego zagadnienia jeszcze powrócę, ale najpierw pokażę, jak używać naszej funkcji.

Z funkcją zdefiniowaną przez użytkownika pracuje się w dokładnie taki sam sposób jak ze zwykłą funkcją. Na listingu 5.4 pokazałem przykład zapytania pobierającego wartości stężenia alkoholu we krwi za pomocą wywołań funkcji w pakiecie.

**Listing 5.4.** *Pobieranie wartości stężenia alkoholu we krwi dla kobiety i mężczyzny*

```

SQL> select
2     p.id as p_id
3     , p.name
4     , pa.sales_volume as vol
5     , pa.abv
6     , formulas.bac(pa.sales_volume, pa.abv, 80, 'M') bac_m
7     , formulas.bac(pa.sales_volume, pa.abv, 60, 'F') bac_f
8  from products p
9  join product_alcohol pa
10     on pa.product_id = p.id
11  where p.group_id = 142
12  order by p.id;

```

Wygenerowane zostają takie same dane wyjściowe jak w przypadku listingu 5.2, więc nie ma tutaj żadnych niespodzianek.

To rozwiązanie jest łatwe w użyciu, ponieważ kod funkcji jest tworzony w zwykły sposób. Jeżeli wiadomo, że dana funkcja będzie wywołana przede wszystkim z poziomu kodu SQL i rzadziej (o ile w ogóle) z PL/SQL, wówczas wystarczy dodać dyrektywę `PRAGMA UDF`, a kompilator zajmie się resztą. Potencjalną korzyścią jest zmniejszenie w trakcie wykonywania kodu obciążenia związanego z przełączaniem kontekstu.

Wielkość korzyści z użycia dyrektywy `PRAGMA UDF` będzie zależała od wielu czynników. Jeżeli kod zdefiniowany w funkcji PL/SQL składa się z poleceń, które *można* wyrazić bezpośrednio za pomocą czystego kodu SQL (takich jak funkcja `formulas.bac`), wówczas korzyści prawdopodobnie będą większe. Natomiast zastosowanie znacznie bardziej skomplikowanych funkcji, zawierających sporo funkcjonalności PL/SQL bądź osadzonego kodu SQL, przynosi tylko niewielkie korzyści lub nie przynosi żadnych. Zawsze należy przeprowadzać odpowiednie testy. Ogólnie rzecz biorąc, dyrektywa `PRAGMA UDF` nie zaszkodzi, a może nieco pomóc, gdy *wiadomo*, że funkcja będzie używana praktycznie wyłącznie z poziomu kodu SQL.

Podczas kompilowania funkcji zdefiniowanej razem z dyrektywą `PRAGMA UDF` staram się możliwie zmniejszyć koszt jej wywołania z poziomu kodu SQL. To jednocześnie oznacza, że nie przejmuję się *potencjalnie* większym kosztem jej wywołania z poziomu PL/SQL. W zależności od wielu czynników może tutaj wystąpić niewielki efekt uboczny, ponieważ funkcja zdefiniowana z `PRAGMA UDF` oczekuje danych w formacie dostarczonym przez silnik SQL. Ten efekt może być niezauważalny, wszystko zależy od konkretnych okoliczności.

Zamiast używać funkcji `PRAGMA UDF`, można pominąć tworzenie funkcji składowej w bazie danych i po prostu podać tę funkcję w samym zapytaniu.

## Zdefiniowanie funkcji za pomocą klauzuli WITH

Wersja 12.1 bazy danych Oracle pozwala umieszczać kod funkcji PL/SQL (a także procedury, co rzadko okazuje się użyteczne) bezpośrednio w klauzuli `WITH` zapytania, jak pokazałem na listingu 5.5.

**Listing 5.5.** Obliczanie stężenia alkoholu we krwi za pomocą klauzuli `WITH`

```
SQL> with
2   function bac (
3       p_volume in number
4       , p_abv   in number
5       , p_weight in number
6       , p_gender in varchar2
7   ) return number deterministic
8   is
9   begin
10      return round(
11          100 * (p_volume * p_abv / 100 * 0.789)
12          / (p_weight * 1000 * case p_gender
13                                  when 'M' then 0.68
14                                  when 'F' then 0.55
15          end)
```

```

16         , 3
17     );
18 end;
19 select
20     p.id as p_id
21     , p.name
22     , pa.sales_volume as vol
23     , pa.abv
24     , bac(pa.sales_volume, pa.abv, 80, 'M') bac_m
25     , bac(pa.sales_volume, pa.abv, 60, 'F') bac_f
26 from products p
27 join product_alcohol pa
28     on pa.product_id = p.id
29 where p.group_id = 142
30 order by p.id
31 /

```

Na początku kodu znajduje się słowo kluczowe `WITH`, podobnie jak w przykładach zamieszczonych w rozdziale 3. Jednak zamiast podzapytania wiersze od 2. do 18. zawierają funkcję `bac()`, podobną do zdefiniowanej w pakiecie `formulas`. Może być ona wywoływana z poziomu kodu SQL, jak pokazałem w wierszach 24. i 25. Dane wyjściowe tego zapytania są identyczne z wygenerowanymi przez kod przedstawiony na listingu 5.2.

Funkcja zdefiniowana w klauzuli `WITH` jest kompilowana w taki sam sposób jak funkcja z dyrektywą `PRAGMA UDF`, przy czym nie jest przechowywana w słowniku danych, jak w przypadku obiektu PL/SQL. Taka funkcja zostanie razem z zapytaniem zapisana jedynie w puli współdzielonej i *nie* może być wywoływana z poziomu innego zapytania SQL lub PL/SQL.

---

■ **Uwaga** Wiersz 31. na listingu 5.5 kończy się ukośnikiem `/` zamiast średnikiem. Gdy analizator składni ustali, że klauzula `WITH` zawiera kod PL/SQL, nie będzie w stanie (przynajmniej obecnie) stwierdzić, czy ukośnik wskazuje koniec polecenia czy też fragment kodu PL/SQL. To może się zmienić w przyszłych wersjach bazy danych. Obecnie rozwiązaniem jest użycie ukośnika, który pozwala wskazać SQLcl lub SQL\*Plus koniec polecenia.

---

Istnieje możliwość zdefiniowania wielu funkcji w pojedynczej klauzuli `WITH`. Na przykład mogą się zdecydować na refaktoryzację kodu i utworzenie dwóch funkcji pomocniczych, przeznaczonych do obliczania gramów alkoholu i gramów płynu w ciecie (wartości wyrażone w mililitrach). Następnie obie funkcje pomocnicze będą używane w funkcji `bac()`. Przykład takiego rozwiązania zaprezentowałem na listingu 5.6, który jest nieco dłuższy od wcześniejszych, ale jednocześnie znacznie bardziej samodokumentujący się.

**Listing 5.6.** Przykład zdefiniowania wielu funkcji w klauzuli `WITH`

```

SQL> with
2     function gram_alcohol (
3         p_volume in number
4         , p_abv   in number
5     ) return number deterministic
6     is
7     begin

```

```

8      return p_volume * p_abv / 100 * 0.789;
9  end;
10  function gram_body_fluid (
11      p_weight in number
12      , p_gender in varchar2
13  ) return number deterministic
14  is
15  begin
16      return p_weight * 1000 * case p_gender
17                                  when 'M' then 0.68
18                                  when 'F' then 0.55
19                                  end;
20  end;
21  function bac (
22      p_volume in number
23      , p_abv   in number
24      , p_weight in number
25      , p_gender in varchar2
26  ) return number deterministic
27  is
28  begin
29      return round(
30          100 * gram_alcohol(p_volume, p_abv)
31          / gram_body_fluid(p_weight, p_gender)
32      , 3
33      );
34  end;
35  select
...

```

Wiele funkcji zdefiniowanych w klauzuli WITH nie ma wpływu na generowane dane wyjściowe — pozostają bez zmian.

Do programisty należy decyzja, czy chce używać pojedynczej funkcji czy ich większej liczby. Jeżeli funkcja ma być wywołana w wielu zapytaniach SQL, wówczas należy utworzyć funkcję składowaną (z dyrektywą PRAGMA UDF lub bez niej), co do tego nie ma żadnych wątpliwości. Natomiast w innych przypadkach może być zasadne pytanie o sens umieszczania funkcji w klauzuli WITH zamiast jej zdefiniowania razem z dyrektywą PRAGMA UDF.

Jednym z takich przypadków jest sytuacja, w której np. nie można zdefiniować funkcji lub procedury składowej, ponieważ baza danych jest w trybie tylko do odczytu bądź tworzone są polecenia i narzędzia przeznaczone do uruchamiania bez konieczności instalowania kodu w bazach danych klientów.

Jeszcze inna sytuacja to taka, w której funkcja w rzadkich przypadkach wykonuje dynamiczny kod SQL i z jakiegokolwiek powodu nie może używać wiązania zmiennych, a zamiast tego do przygotowania kodu SQL korzysta z konkatenacji ciągu tekstowego. Zdefiniowanie funkcji w zapytaniu zapewnia pełną kontrolę nad argumentami używanymi podczas jej wywołania, co z kolei pozwala zabezpieczyć się przed atakami typu SQL injection. Taka funkcja nie będzie mogła być wywołana z innego miejsca.

Trzeci przypadek zachodzi wtedy, gdy funkcjonalność jest ściśle związana z pojedynczym celem, i wówczas można zdecydować się na inny sposób hermetyzacji kodu.

## Hermetyzacja kodu w widoku

W przypadku omawianej tutaj aplikacji rozsądne jest stwierdzenie, że obliczanie stężenia alkoholu we krwi nie ma sensu poza kontekstem rekordu w tabeli `product_alcohol`. Uciekając się do analogii programowania zorientowanego obiektowo, można by powiedzieć, że to jest metoda *egzemplarza*, a nie *statyczna*.

Podobny efekt można osiągnąć przez utworzenie widoku, jak pokazałem na listingu 5.7.

**Listing 5.7.** Tworzenie widoku przeznaczonego do obliczania stężenia alkoholu we krwi

```
SQL> create view product_alcohol_bac
 2 as
 3 with
 4     function gram_alcohol (
...
12     function gram_body_fluid (
...
23     function bac (
...
37 select
38     pa.product_id
39     , pa.sales_volume
40     , pa.abv
41     , bac(pa.sales_volume, pa.abv, 80, 'M') bac_m
42     , bac(pa.sales_volume, pa.abv, 60, 'F') bac_f
43 from product_alcohol pa
44 /
```

View PRODUCT\_ALCOHOL\_BAC created.

W tym widoku użyłem klauzuli `WITH` razem z trzema funkcjami pochodzącymi z listingu 5.6. Zapytanie zostało zdefiniowane w wierszach od 37. do 43. — używa jedynie tabeli `product_alcohol` i pobiera z niej wszystkie kolumny plus wartości dwóch obliczonych kolumn, `bac_m` i `bac_f`.

Teraz można wykonać zapytanie złączające tabelę `products` z widokiem `product_alcohol_bac`, jak pokazałem na listingu 5.8. W ten sposób żądane dane pobieramy bezpośrednio i bez problemów.

**Listing 5.8.** Pobieranie za pomocą widoku danych dotyczących stężenia alkoholu we krwi

```
SQL> select
 2     p.id as p_id
 3     , p.name
 4     , pab.sales_volume as vol
 5     , pab.abv
 6     , pab.bac_m
 7     , pab.bac_f
 8 from products p
 9 join product_alcohol_bac pab
10     on pab.product_id = p.id
11 where p.group_id = 142
12 order by p.id;
```

Otrzymujemy te same dane wyjściowe.

| P_ID | NAME             | VOL | ABV | BAC_M | BAC_F |
|------|------------------|-----|-----|-------|-------|
| ---- | -----            | --- | --- | ----- | ----- |
| 4040 | Coalminers Sweat | 330 | 8.5 | 0.041 | 0.067 |
| 4160 | Reindeer Fuel    | 500 | 6   | 0.044 | 0.072 |
| 4280 | Hoppy Crude Oil  | 330 | 7   | 0.034 | 0.055 |

Ta metoda pozwala wielokrotnie używać logiki przez wykonywanie zapytań SQL do widoku zamiast do tabeli. Mimo to logika będzie znajdowała się w pojedynczym miejscu — w definicji widoku.

Zamiast definiować funkcje w widoku, ten sam efekt można osiągnąć, gdy wynik wywołuje funkcję pakietu `formulas.bac()`. Jeżeli jednak funkcjonalność jest *ściśle* związana jedynie z określonym zapytaniem lub definicją widoku, wówczas lepsze będzie przechowywanie kodu w jednym miejscu i niezaśmiecanie słownika danych funkcjami składowymi, które i tak nigdy nie powinny być wywoływane poza określonym kodem SQL.

## Podsumowanie

Wprawdzie tematem tej książki nie jest PL/SQL jako taki, ale możliwość jeszcze ściślejszej niż wcześniej integracji kodu PL/SQL i SQL jest funkcjonalnością, która powinna być znana programistom SQL. Dzięki informacjom zamieszczonym w rozdziale potrafisz:

- Ustalać, czy funkcja jest wywołana przede wszystkim z poziomu SQL. W takim przypadku korzystne będzie dodanie dyrektywy `PRAGMA UDF` do jej definicji.
- Osadzać funkcje „jednorazowe” w zapytaniach SQL z użyciem klauzuli `WITH`.
- Sprawdzać, czy konkretna funkcjonalność może działać lepiej po jej hermetyzacji w widoku za pomocą klauzuli `WITH` zamiast w tradycyjnych funkcjach składowanych.

Podczas codziennej pracy prawdopodobnie najczęściej będziesz korzystać z funkcji zawierających dyrektywy `PRAGMA UDF`. Mimo to technika oparta na klauzuli `WITH` również może być niezwykle użyteczna, zwłaszcza w sytuacjach, w których nie można instalować funkcji lub procedur składowanych.





# Skorowidz

## A

- agregacja, 184, 232
  - ciągu tekstowego, 148
- akumulacja, 253
- algorytm
  - MFFD, 334
  - optymalizacji pakowania, 329, 335
  - przybliżonego najlepszego dopasowania, 334
- aliasy, 37
  - kolumn, 59
- analiza
  - cykli pobierania produktów, 229
  - czynności pobierania produktów, 227
  - danych, 190
  - dopasowania wzorca, 275
  - efektywności pracownika, 226
  - przyjazdów i odjazdów, 223
- ANSI SQL, 131
- antyłączenia, 45
- anulowanie przedstawienia kolumn, unpivoting, 99–112
  - dynamiczne, 104
  - samodzielne, 102

## B

- błąd, 65, 151, 152
  - ORA-00918, 122
  - ORA-01489, 158

## C

- ciąg tekstowy
  - agregacja, 148
  - jako tablica JSON, 135
  - konkatenacja, 148
  - konwersja kolekcji, 150
  - wyodrębnienie, 156
- cykl odjazd-przyjazd-odjazd, 224
- cykle
  - analiza, 229
  - dane statystyczne, 231
  - grupowanie, 230
  - identyfikowanie, 228, 232
- czynnik sezonowości, 240

## D

- dane statystyczne
  - cykle, 231
  - efektywność pracownika, 226
  - lista produktów, 233
- definiowanie
  - kolejności, 172
  - okna, 175
- diagramy Venna, 32
- dopasowanie
  - V, 273
  - W, 276
  - wzorca, 350
  - wzorca rekordu, 232, 261, 263

drzewo, 349  
 dynamiczne  
   anulowanie przestawiania kolumn, 112  
   mapowanie tabeli wymiaru, 111  
 dynamiczny SQL, 71  
 działanie klauzuli PIVOT, 120  
 dziennik zdarzeń, 220  
   oś czasu, 223  
   pobierania produktów, 221  
   przyjazdy i odjazdy, 223

## F

faktoryzacja podzapytania, 49  
 FIFO, first in, first out, 198, 201, 205, 210, 215, 217  
 filtrowanie  
   według pseudokolumny, 65, 186  
   według zakumulowanej sumy, 204  
   za pomocą definicji wzorca, 358  
 format JSON, 142  
 funkcja  
   APEX, 132  
   apex\_string.join\_clob(), 161, 162  
   apex\_string.split(), 133, 136, 140  
   avg(), 53  
   cast(), 151  
   classifier(), 266, 344, 353  
   collect(), 150, 151, 155  
   count(\*), 231  
   delimited\_cow\_row.parser(), 138  
   dense\_rank(), 187, 193, 195, 207, 219  
   evaluate\_expr(), 72  
   extract(), 157  
   first(), 270  
   formulas.bac(), 83  
   getclobcal(), 161  
   getstringval(), 161  
   json\_arrayagg(), 161  
   json\_table(), 135, 143, 144  
   last(), 270  
   last\_value(), 228, 229  
   lead(), 73, 220, 223–226, 229, 291, 319  
   listagg(), 90, 148, 155, 158, 160, 162  
   match\_number(), 266, 353, 356  
   multiset(), 46  
   name\_coll\_type\_to\_clob(), 160  
   name\_coll\_type\_to\_varchar2(), 150, 151

next(), 317, 328  
 ntile(), 51  
 nullif(), 241  
 nullify(), 242  
 obliczająca stężenie alkoholu, 77  
 ODCI, 137  
 ODCIAggregateInitialize(), 154  
 ODCIAggregateIterate(), 154  
 ODCIAggregateMerge(), 155  
 ODCIAggregateTerminate(), 154  
 odcitableclose(), 139  
 odcitabledescribe(), 139  
 odcitablefetch(), 139  
 odcitableprepare(), 139  
 odcitablestart(), 139  
 parser(), 138  
 pattern(), 288, 292  
 PL/SQL, 71, 77  
 prev(), 328  
 rank(), 187, 193, 195, 196  
 regexp\_substr(), 141  
 regr\_intercept(), 244, 248  
 regr\_slope(), 244, 248  
 replace(), 143  
 row\_number(), 47, 185, 187, 193, 194  
 rtrim(), 157  
 stragg(), 152, 153, 155  
 substr(), 140  
 sum(), 90, 168, 218  
 table(), 46, 131, 133, 135  
 xmlagg(), 156, 160  
 xmlelement(), 157  
 xmlparse(), 157  
 xmltable(), 145

funkcje  
   agregacji, 148, 353  
   analityczne, 24, 163  
   potokowane, 130  
   statystyczne, 226  
   tabeli, 28  
   w klauzuli WITH, 79, 80

## G

generowanie rekordów, 133  
   liczby, 134, 141  
   wymiaru, 103

gra w życie, 85  
 generacje, 91, 94  
 liczba żywych sąsiadów, 89  
 siatka, 86, 96  
 tworzenie generacji zerowej, 86  
 wizualizacja generacji zerowej, 87

grupowanie, 70, 283  
 cykli, 230  
 klauzula MATCH\_RECOGNIZE, 287, 290  
 kolejnych danych, 284  
 metoda Tabibitosan, 286  
 określona przerwa, 293  
 ustalona granica, 295  
 warunki, 284  
 wykrywanie przerw, 291, 292

## H

hermetyzacja kodu, 82  
 hierarchiczne drzewo pracowników, 347

## I

implementowanie funkcji ODCI, 137  
 iteracje, 91, 257

## K

klasyfikacja  
 down, 265, 267, 279  
 high, 326  
 higher, 351  
 low, 326  
 medium, 326  
 one\_higher, 288  
 same, 267  
 SAME, 273  
 strt, 268  
 up, 265, 279  
 klauzula  
 AFTER MATCH, 272, 279, 353  
 APPLY, 138  
 CONNECT BY, 64, 65, 67, 133  
 CROSS APPLY, 27, 28  
 CROSS JOIN, 26  
 DEFINE, 265, 294  
 DIMENSION BY, 88  
 DISTINCT, 159

EXECUTE IMMEDIATE, 72  
 FETCH FIRST, 189, 196  
 FOR, 125  
 FROM DUAL, 60  
 GROUP BY, 122, 283  
 IGNORE NULLS, 227  
 IN, 210  
 INSERT INTO ... SELECT ..., 248  
 LATERAL, 28  
 LEFT OUTER JOIN LATERAL, 29  
 MATCH\_RECOGNIZE, 232, 263, 284, 287, 290, 322, 332, 346, 350, 359  
 MEASURES, 88, 270, 332  
 MODEL, 86, 88, 258  
 ON 1=1, 27  
 ORDER BY, 23, 37, 149, 170, 205, 265  
 OUTER APPLY, 28, 29, 144  
 OUTER JOIN, 29  
 OUTER JOIN LATERAL, 29  
 PARTITION BY, 179, 194, 265, 336, 345  
 PATTERN, 269, 343  
 PERIOD FOR, 302  
 PIVOT, 119, 194, 230, 355  
 RULES, 259  
 SELECT, 37, 39, 119, 200, 204  
 SELECT FROM DUAL, 102  
 START WITH, 63, 65  
 UNPIVOT, 101, 102, 105–108, 112  
 WHERE, 51, 203, 204  
 WITH, 53, 57, 61, 79, 215, 353

klauzule  
 analityczne, 167  
 ograniczające, 192  
 okna, 172  
 klienci SQL, 40  
 klucz-wartość, *Patrz* wymiar, miara, 104  
 kolejność FIFO, 198, 201, 205, 210, 215, 217  
 kolekcja  
 konwersja na ciąg, 150  
 komunikat o błędzie, 123, 158  
 konkatencja, 24  
 ciągu tekstowego, 148  
 zbioru, 36  
 konwersja  
 ciągu tekstowego, 142  
 kolekcji, 150  
 na format JSON, 142  
 rekordów na kolumny, 119

**L**

licznik odwiedzin, 313  
 dzienny, 319  
 historyczne dane, 314  
 skoki wzrostu, 317  
 wyświetlenie danych, 314  
 linia trendu, 244  
 lista, 148  
 nazw kolumn, 59

**Ł**

łączenie zakresów dat, 298

**M**

maszyna wirtualna Database  
 App Development VM, 17  
 metoda Tabibitosan, 285, 286, 289  
 MFFD, modified first fit decreasing, 334  
 miara, 104  
 mnożenie ilości hierarchicznych, 64  
 modularyzacja zapytania SQL, 49

**N**

Najlepsze-N rekordów, 181  
 w wielu partycjach, 194  
 nawias kwadratowy, 135  
 nazwane podzapytania, 49, 54  
 niejawne  
 grupowanie, 120  
 struktury case, 126  
 niestandardowa funkcja tabeli, 137

**O**

obliczanie  
 czynnika sezonowości, 240  
 liczby dziennych odwiedzin, 322  
 oczekiwanej wielkości stanów  
 magazynowych, 254  
 przedziałów ilościowych, 212, 213  
 spadku stanów magazynowych, 255  
 stężenia alkoholu we krwi, 76, 79, 82  
 sumy, 167, 170, 179  
 trendu, 243  
 wielkości stanów magazynowych, 254  
 wyśrodkowanej średniej ruchomej, 239

obsługa wartości null, 310  
 ODCI, Oracle Data Cartridge Interface, 137, 152  
 okno

domyślne, 176  
 rekordów, 177  
 zakresu, 177

okresy zatrudnienia  
 wykres, 301  
 wyświetlanie, 300  
 złączanie, 303

opcja

rows only, 189  
 rows with ties, 189  
 with ties, 191, 192

operacje na zbiorach, 32

operator

EXCEPT, 45  
 MINUS, 45  
 MINUS ALL, 46, 47  
 MULTISSET EXCEPT, 44  
 MULTISSET EXCEPT ALL, 44, 46  
 MULTISSET EXCEPT DISTINCT, 44  
 MULTISSET INTERSECT, 43  
 MULTISSET INTERSECT DISTINCT, 43  
 MULTISSET UNION, 41  
 MULTISSET UNION ALL, 41  
 MULTISSET UNION DISTINCT, 41  
 UNION, 38  
 UNION ALL, 36, 39

operatory

wielozbioru, 33, 40  
 zbioru, 34, 38

optymalizacja pakowania, 329, 331, 338, 340, 346

**P**

pakiet

apex\_data\_parser, 144  
 dbms\_xmlindex, 152

pakowanie

optymalizacja, 329, 331, 338, 340, 346

partycje, 168

partycjonowanie, 172, 335

plan wykonania zapytania, 57

planowanie miesięcznej sprzedaży, 252

pliki kodu źródłowego, 15

pobieranie  
   danych, 200  
   danych dzienników zdarzeń, 221  
 produktów  
   analiza cykli, 229  
   dziennik zdarzeń, 221  
   identyfikowanie cykli, 232  
   kolejność FIFO, 205, 217  
   optymalna trasa, 201, 207  
   partiami, 210  
   przedziały ilościowe, 211, 212  
   rekordów, 140  
 podzapytania  
   faktoryzacja rekurencyjna, 62, 66  
   fifo, 215, 218  
   nazwane, 54  
   o takiej samej nazwie, 60  
   skalarne, 23, 89, 349  
   wielokrotne użycie, 55  
 podzapytanie  
   olines, 215, 218  
   orderbatch, 211, 215, 218  
   pick, 218  
 podział ciągu tekstowego, 133  
 połączenie wielu wymiarów i miar, 124  
 porównywanie wartości rekordów, 304, 306  
 potokowana funkcja tabeli, 129  
 PRAGMA UDF, 77  
 prognozowanie, 236, 243, 244, 246, 249  
 przedziały ilościowe, 211  
   obliczanie, 212, 213  
   złączenie dla produktów i zamówień, 214  
 przestawianie  
   column, pivoting, 99, 119  
   ręczne, 121  
   rekordów, 116  
 przetwarzanie  
   danych JSON, 143  
   twarde, hard parse, 137, 139  
 pseudokolumna, 139, 186

## R

ranking, 187  
 refaktoryzacja, 54  
 regresja  
   liniowa, 235, 242  
   punkt wyjścia, 239

reguła  
   najlepszych rekordów, 182  
   najwyższych wartości, 183  
   olimpijska, 183  
 rekurencyjna faktoryzacja podzapytania, 58, 62, 66, 256, 260  
   mnożenie ilości hierarchicznych, 66  
   wyszukiwanie liści drzewa, 68  
 relacje hierarchiczne, 63  
 remis, 190, 192  
 rozkład produktów, 342

## S

schemat  
   APEX, 133  
   practical, 17  
 składnia  
   after match skip to, 281  
   after match skip to next row, 352, 359  
   all rows, 265  
   all rows per match, 316  
   CROSS APPLY, 131  
   funkcji analitycznej, 167  
   klauzula\_okna, 170  
   klauzuli PATTERN, 269  
   klauzuli\_order\_by, 170  
   one row per match, 270, 353, 357  
   rows between, 178  
   rows between unbounded preceding and current row, 354  
   to rows between unbounded preceding and unbounded following, 354  
   with unmatched rows, 340  
 słowo kluczowe  
   ALL, 39, 42  
   asc, 181  
   current, 176  
   desc, 181  
   distinct, 159, 160  
   DISTINCT, 41, 42  
   down, 267  
   final, 354  
   LATERAL, 30  
   on overflow, 159  
   parallel\_enable, 155  
   PIPELINED, 129  
   PIVOT, 119

## słowo kluczowe

- range, 176
- same, 267
- siblings, 169
- TABLE, 130
- truncate, 159
- up, 267
- value, 176
- view, 82
- WITH, 80
- within, 149
- stany magazynowe, 249, 251
  - agregacja, 250
  - obliczanie
    - spadku, 255
    - wielkości, 254
  - uzupełnianie, 255, 256, 258
- struktura drzewa, 347, 349
- suma krocząca, 198, 249
- sumowanie, 179
- szeregi czasowe, 237

**T**

## tabele

- json\_table(), 145
- tworzenie, 302
- wymiarów, 109
  - anulowanie przestawienia kolumn, 109
  - dynamiczne mapowanie, 111
- z ważnością czasową, 302
- zagnieżdżone, 47, 130

## tablica JSON, 135

## trend, 243

## tworzenie

- listy, 148
- siatki, 86
- szeregów czasowych, 237
- tabeli, 302
- widoku, 82

## typ

- clob, 160
- kolekcji, 129
- name\_coll\_type, 150
- stragg\_type, 154
- varchar2, 158

**U**

UDF, user-defined function, 78

**V**

VirtualBox, 17

**W**

wartość null, 29, 149, 173, 302, 310

warunek case, 258

## widok

- brewery\_products, 147
- emp\_hire\_periods\_with\_name, 299
- inventory\_with\_dims, 200
- inventory\_totals, 250
- monthly\_orders, 251
- product\_alcohol\_bac, 82
- total\_sales, 184
- web\_page\_counter\_hist, 313
- yearly\_sales, 185

## widoki

- osadzone, 22, 89, 103, 160, 212, 237
- skorelowane, 26, 28
- przeznaczone do agregacji, 184

## wielozbiory, 40

## wykres

- codziennych odwiedzin strony, 320
- historii licznika odwiedzin, 315
- linii trendu, 244
- okresów zatrudnienia, 301
- sprzedaży, 238

## wykrywanie przerw, 291, 292

## wymiar, 104

## wyrażenia regularne, 269

## wyrażenie case, 229

## wyszukiwanie

- dopasowania, 269, 272, 276
- liści drzewa, 68
- nagłych skoków, 326, 327
- okresów
  - według wartości względnej, 323, 324
- wyśrodkowana średnia ruchoma, 239, 240
- wyświetlanie
  - danych licznika, 314
  - nazw kolumn, 58

wzorzec  
  codzienne odwiedziny strony, 320  
  grupowanie danych, 283  
  wyszukiwanie, 263  
  wyszukiwanie nagłych skoków, 326  
wzór Widmarka, 76, 77

## Z

zagnieżdżone widoki, 54  
zapytania  
  hierarchiczne, 347  
  planowanie, 57  
zapytanie, *Patrz* klauzula  
zapytanie\_partycjonujące, 168  
zbiory, 32  
  iloczyn, 32, 38  
  konkatenacja, 36  
  unia, 32, 38  
  wykluczenie, 32, 38

zliczanie, count, 349  
  dopasowanie wzorca, 350  
  rekordów  
  rzeczywistych kolumn, 139  
złączenia, 26  
  w stylu ANSI, 131  
złączenie  
  CROSS JOIN LATERAL, 134  
  LATERAL, 196  
  LEFT OUTER JOIN, 131, 237  
  orderlines, 201  
  typu OUTER, 28, 30  
zmienna wiązania, 114  
znak |, 265



# PROGRAM PARTNERSKI

— GRUPY HELION —



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

**Dowiedz się więcej i dołącz już dzisiaj!**

<http://program-partnerski.helion.pl>

GRUPA  
**Helion**

# SQL. Opanuj sztukę pisania naprawdę zaawansowanych zapytań!

Mimo upływu lat SQL jest bezkonkurencyjnym narzędziem do przetwarzania danych. Bazy danych Oracle wciąż imponują możliwościami. W ciągu ostatnich dekad bowiem obie te technologie były konsekwentnie unowocześniane i usprawniane. W efekcie nawet za pomocą jednego, choć niekiedy dość złożonego zapytania SQL można przeprowadzić operacje, które w innym języku wymagałyby napisania długich bloków kodu. Jednak do uzyskania spektakularnej wydajności i szybkości aplikacji niezbędne jest duże doświadczenie w posługiwaniu się zaawansowanymi konstrukcjami SQL.

To książka przeznaczona dla osób, które dobrze poznały podstawy języka SQL i chcą nabrać biegłości w praktycznym zastosowaniu jego zaawansowanych funkcji. Poszczególne zagadnienia zostały zaprezentowane poprzez stopniową rozbudowę i zwiększanie złożoności prostych zapytań SQL. Omówiono takie techniki jak korelacja widoku osadzonego, operacje na zbiorach, analiza dzienników zdarzeń, a także sposoby używania klauzul, między innymi MODEL czy MATCH\_RECOGNIZE. Znalazło się tu mnóstwo przykładów kodu SQL, skonstruowanego tak, aby maksymalnie ułatwić zrozumienie prezentowanych treści. To pomoże Ci zdobyć umiejętności, dzięki którym wydajność i wygoda użytkowania Twoich aplikacji istotnie się zwiększą!

W książce:

- stosowanie zaawansowanych funkcji języka SQL w bazie danych Oracle
- stopniowe usprawnianie zapytań SQL
- przetwarzanie większej ilości danych za pomocą mniejszej liczby zapytań
- korzystanie z funkcji analitycznych
- dopasowywanie wzorca rekordu
- rekurencyjna faktoryzacja podzapytania

**Kim Berg Hansen** jest duńskim programistą baz danych. Od ponad 20 lat zajmuje się technologiami Oracle SQL i PL/SQL. Podczas pisania kodu zwraca baczność na jego efektywność i niskie wykorzystanie zasobów komputera, a także na uzyskanie najlepszych wrażeń użytkownika gotowej aplikacji. Posiada tytuły Oracle Certified Expert (OCE) w zakresie SQL i Oracle ACE Director.

|                                                                                                                                                                                                                                                                                             |                                                                                                                                                |                                                                                                                                  |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|
|                                                                                                                                                                                                           | Sprawdź nasze szkolenia!                                                                                                                       | KOD KORZYŚCI<br>Sięgnij po więcej!            |
|  <b>helion.pl</b><br> <b>HELION SA</b><br>ul. Kościuszki 1c<br>44-100 Gliwice<br>tel.: 32 230 98 63<br>helion@helion.pl | <br><b>AKADEMIA IT &amp; BUSINESS</b><br>HELIONSZKOLENIA.PL | ISBN 978-83-283-8733-1<br><br>9 788328 387331 |
| <b>INFORMATYKA W NAJLEPSZYM WYDANIU</b>                                                                                                                                                                                                                                                     |                                                                                                                                                | Cena: 77,00 zł                                                                                                                   |

Apress®