

Pierwsze kroki z SQL

PRAKTYCZNE PODEJŚCIE
DLA POCZĄTKUJĄCYCH



Tytuł oryginału: Getting Started with SQL: A Hands-On Approach for Beginners

Tłumaczenie: Beata Błaszczyk

ISBN: 978-83-283-2818-1

© 2017 Helion S.A.

Authorized Polish translation of the English edition of Getting Started with SQL, ISBN 9781491938614 © 2016 Thomas Nield

This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

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 Wydawnictwo HELION 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 Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
<ftp://ftp.helion.pl/przyklady/pksqlp.zip>

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
<http://helion.pl/user/opinie/pksqlp>
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

Spis treści

Przedmowa.....	7
Wstęp	9
1. Czym jest język SQL i dlaczego jest taki popularny?.....	15
Kto powinien znać język SQL?	16
2. Bazy danych	17
Czym jest baza danych?	17
Podstawowe informacje o relacyjnych bazach danych	17
Dlaczego konieczne jest stosowanie oddzielnych tabel?	18
Wybór odpowiedniej bazy danych	19
3. SQLite	23
Czym jest SQLite?	23
SQLiteStudio	24
Dodawanie bazy danych i podgląd jej zawartości	25
4. Polecenie SELECT	31
Wyszukiwanie danych za pomocą języka SQL	31
Wyrażenia w instrukcjach SELECT	34
Konkatenacja pól tekstowych	39
Podsumowanie	40

5. Klauzula WHERE	41
Filtrowanie rekordów	41
Korzystanie z klauzuli WHERE	
w odniesieniu do wartości liczbowych	42
Operatory AND, OR oraz IN	43
Klauzula WHERE a wyrażenia tekstowe	45
Klauzula WHERE a wartości logiczne	47
Obsługa wartości NULL	48
Warunki grupujące	50
Podsumowanie	52
6. Klauzule GROUP BY i ORDER BY	53
Grupowanie wierszy	53
Sortowanie wierszy	56
Funkcje agregujące	56
Klauzula HAVING	59
Wyświetlanie unikalnych wierszy	61
Podsumowanie	61
7. Instrukcje ze słowem kluczowym CASE	63
Składnia instrukcji CASE	63
Grupowanie w instrukcji CASE	65
Trik z wartością 0 i null w instrukcji CASE	65
Podsumowanie	68
8. Operator JOIN	71
Łączenie ze sobą dwóch tabel	71
Złączenie wewnętrzne (INNER JOIN)	74
Złączenie lewostronne (LEFT JOIN)	77
Inne rodzaje złączeń	80
Łączenie ze sobą wielu tabel	81
Grupowanie w instrukcjach ze złączeniami	83
Podsumowanie	86

9. Projektowanie baz danych	87
Planowanie projektu bazy danych	87
Baza danych SurgeTech Conference	90
Klucze główne i obce	92
Schemat bazy danych	94
Tworzenie nowej bazy danych	95
Instrukcja CREATE TABLE	97
Definiowanie kluczy obcych	106
Tworzenie widoków	108
Podsumowanie	111
10. Zarządzanie danymi	113
Instrukcja INSERT	114
Instrukcja DELETE	117
Instrukcja TRUNCATE TABLE	118
Instrukcja UPDATE	119
Instrukcja DROP TABLE	119
Podsumowanie	120
11. Dalsze kroki w świecie języka SQL	121
A Operatory i funkcje	125
B Informacje dodatkowe	133
Skorowidz	139

Bazy danych

Czym jest baza danych?

Najszerzej rzecz ujmując, **baza danych** to miejsce, w którym dane są gromadzone i porządkowane. Bazą danych jest zarówno arkusz kalkulacyjny zawierający informacje o rezerwacjach klientów, jak i zwykły plik tekstowy z rozkładem lotów. Plik taki może być zapisywany w różnych formatach, takich jak na przykład XML czy CSV.

Gdy eksperci branży IT używają pojęcia *baza danych*, zazwyczaj mają na myśli **system zarządzania relacyjną bazą danych** (ang. *Relational Database Management System* — RDBMS). Określenie to może brzmieć dość technicznie i wzbudzać pewien niepokój, jednak system zarządzania bazą danych to nic innego jak pewnego rodzaju baza danych, zawierająca jedną lub wiele tabel, które mogą być ze sobą powiązane.

Podstawowe informacje o relacyjnych bazach danych

Na pewno wiesz, jak wygląda tabela. Składa się z kolumn i wierszy, w których przechowywane są dane, podobnie jak ma to miejsce w arkuszu kalkulacyjnym. Tabele mogą być ze sobą powiązane. Może na przykład istnieć relacja między tabelą `CUSTOMER_ORDER`¹ a tabelą `CUSTOMER`, do której się ona odwołuje w celu uzyskania informacji o samych klientach.

¹ Tabela o nazwie `CUSTOMER_ORDER` w przykładowej bazie danych, do której będzie nawiązywać większość przykładów zawartych w tej książce, przechowuje informacje o zamówieniach klientów — *przyp. tłum.*

Powiedzmy zatem, że mamy do dyspozycji tabelę `CUSTOMER_ORDER`, w której znajduje się pole o nazwie `CUSTOMER_ID`, jak pokazano na rysunku 2.1.

	ORDER_ID	ORDER_DATE	SHIP_DATE	CUSTOMER_ID	PRODUCT_ID	ORDER_QTY	SHIPPED
1	1	2015-05-15	2015-05-18	1	1	450	false
2	2	2015-05-18	2015-05-21	3	2	600	false
3	3	2015-05-20	2015-05-23	3	5	300	false
4	4	2015-05-18	2015-05-22	5	4	375	false
5	5	2015-05-17	2015-05-20	3	2	500	false

Rysunek 2.1. Tabela `CUSTOMER_ORDER`, zawierająca kolumnę `CUSTOMER_ID`

W naszej bazie danych znajduje się zapewne także inna tabela, prawdopodobnie o nazwie `CUSTOMER`, przedstawiona na rysunku 2.2. Zawiera ona przypuszczalnie szczegółowe informacje o każdym kliencie, do którego przypisana jest odpowiednia wartość w kolumnie `CUSTOMER_ID`.

	CUSTOMER_ID	NAME	REGION	STREET_ADDRESS	CITY	STATE	ZIP
1	1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014
2	2	Rex Tooling Inc	Southwest	6129 Collie Blvd	Dallas	TX	75201
3	3	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032
4	4	Prairie Construction	Southwest	264 Long Rd	Moore	OK	62104
5	5	Marsh Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782

Rysunek 2.2. Tabela `CUSTOMER`

Mając do dyspozycji pole `CUSTOMER_ID` w tabeli `CUSTOMER_ORDER`, możemy poszukiwać informacji o klientach zamieszczonych w tabeli `CUSTOMER`. Taka właśnie jest podstawowa zasada działania relacyjnej bazy danych. W przypadku tego rodzaju bazy danych w tabelach mogą znajdować się pola, które wskazują na informacje zawarte w innych tabelach. Taki sposób działania prawdopodobnie nie jest Ci obcy, jeśli korzystałeś już z funkcji Excela `WYSZUKAJ.PIONOWO` w celu wyświetlenia w danym arkuszu informacji, które pochodzą z innego arkusza w skoroszycie.

Dlaczego konieczne jest stosowanie oddzielnych tabel?

Można zadać sobie następujące pytanie: „Dlaczego dane w relacyjnej bazie danych są przechowywane w oddzielnych tabelach skonstruowanych w taki właśnie sposób?”. Odpowiedzią na nie jest **normalizacja**, zgodnie z którą różne rodzaje danych należy umieszczać w oddzielnych tabelach, nie zaś składować je wszystkie

w jednym miejscu. Gdybyśmy przechowywali wszystkie informacje w jednej tabeli, dane by się powtarzały, byłoby ich zbyt wiele, a ich utrzymanie byłoby bardzo trudne. Wyobraźmy sobie, że wszystkie informacje o klientach znajdują się w tabeli CUSTOMER_ORDER, przedstawionej na rysunku 2.3.

	NAME	REGION	STREET_ADDRESS	CITY	STATE	ZIP	ORDER_ID	ORDER_DATE	SHIP_DATE	ORDER_QTY	SHIPPED
1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014	1	2015-05-15	2015-05-18	450	false
2	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032	2	2015-05-18	2015-05-21	600	false
3	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032	3	2015-05-20	2015-05-23	300	false
4	Marsh Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782	4	2015-05-18	2015-05-22	375	false
5	Re-Barre Construction	Southwest	9043 Windy Dr	Irving	TX	75032	5	2015-05-17	2015-05-20	500	false

Rysunek 2.3. Tabela CUSTOMER_ORDER zawierająca nieznormalizowane dane

Zauważ, że w przypadku zamówień złożonych przez firmę Re-Barre Construc- tion konieczne było powielenie informacji o tym kliencie w trzech oddzielnych wierszach (powtórzona została nazwa klienta, region, adres, miasto, stan i kod pocztowy) — po jednym dla każdego zamówienia. Są one zatem nadmiarowe, niepotrzebnie zabierają miejsce na dysku i są trudne w utrzymaniu. Dlaczego? Wyobraź sobie, że klient zmienił adres. W takim przypadku w powyższej tabeli konieczne by było zmodyfikowanie informacji o wszystkich trzech zamówieniach. Dlatego właśnie lepiej jest przechowywać informacje o klientach i ich zamówie- niach w dwóch oddzielnych tabelach: CUSTOMER i CUSTOMER_ORDER. Wówczas zmiana adresu klienta będzie skutkowałą modyfikacją tylko jednego rekordu w tabeli CUSTOMER, jak pokazano na rysunku 2.4.

	CUSTOMER_ID	NAME	REGION	STREET_ADDRESS	CITY	STATE	ZIP
1	1	LITE Industrial	Southwest	729 Ravine Way	Irving	TX	75014
2	2	Rex Tooling Inc	Southwest	6129 Collie Blvd	Dallas	TX	75201
3	3	Re-Barre Construction	Southwest	10917 Lond Way Rd	Irving	TX	75032
4	4	Prairie Construction	Southwest	264 Long Rd	Moore	OK	62104
5	5	Marsh Lane Metal Works	Southeast	9143 Marsh Ln	Avondale	LA	79782

Rysunek 2.4. Tabela zawierająca znormalizowane dane

W rozdziale 8. przyjrzymy się rodzajom złączeń między tabelami i wykorzystaniu do tego celu operatora JOIN. Operator ten umożliwia wyświetlenie danych z dwóch tabel za pomocą jednego zapytania, a więc na przykład przejrzanie listy klientów wraz z zamówieniami, które złożyli.

Wybór odpowiedniej bazy danych

Korzystanie z relacyjnych baz danych i języka SQL nie jest w żaden sposób ogra- niczone. Należy mieć jednak na uwadze fakt, że istnieje kilka firm i społeczności,

które opracowały własne programy do zarządzania relacyjną bazą danych. W każdym z nich wykorzystywane są tabele, które są odpytywane za pomocą języka SQL. Niektóre z tych rozwiązań bazodanowych są **lekkie** (ang. *lightweight*) i proste w użyciu. Dane są wówczas przechowywane w pojedynczym pliku i dostęp do nich ma niewielu użytkowników. Inne rozwiązania są natomiast bardziej rozbudowane i do działania potrzebują serwera. W takim przypadku z danych korzystają jednocześnie tysiące użytkowników i aplikacji. Są również dostępne zarówno darmowe rozwiązania bazodanowe typu *open source*, jak i takie, do korzystania z których wymagane jest zakupienie licencji komercyjnych.

Ze względów praktycznych podzielimy zatem rozwiązania bazodanowe na dwie kategorie: *lekkie* i **scentralizowane** (ang. *centralized*). Chociaż w branży IT taka nomenklatura nie jest zbyt często stosowana, to wyodrębnienie tych dwóch rodzajów baz danych pomoże w objaśnieniu różnic występujących między nimi.

Lekkie bazy danych

Jeśli szukasz prostego rozwiązania, z którego korzystać będzie jeden lub zaledwie kilku użytkowników (np. Twój współpracownicy), zacznij od gromadzenia danych w lekkiej bazie danych. Tego typu bazy nie wymagają bowiem dodatkowych nakładów w postaci serwera. Są również bardzo elastyczne. Dane są z reguły przechowywane w pliku, który można współdzielić z innymi osobami. To jednak stwarza niebezpieczeństwo, że w przypadku wykonywania w pliku zmian jednocześnie przez kilku użytkowników, przechowywane w nim dane utracą spójność. Jeśli zatem przyjdzie Ci się zmierzyć z tym problemem, będzie to oznaczać, że czas rozważyć migrację danych do scentralizowanej bazy danych.

Najczęściej wykorzystywane lekkie bazy danych to SQLite oraz Microsoft Access. W tej książce będziemy korzystać z SQLite. Jest to darmowa, lekka baza danych, której obsługa jest bardzo intuicyjna. Jest ona wykorzystywana w większości urządzeń, z którymi mamy styczność na co dzień. Jest elementem oprogramowania smartfonów, satelitów, samolotów i systemów samochodowych. Ma praktycznie nieograniczoną pojemność i jest idealnym rozwiązaniem w przypadku urządzeń, z których korzysta więcej niż jedna osoba (lub co najwyżej kilka osób). Ze względu na łatwość instalacji tej bazy danych oraz jej prostotę SQLite jest również idealnym środowiskiem do nauki języka SQL.

Microsoft Access jest dostępny na rynku już od jakiegoś czasu, jednak w kontekście skalowalności i wydajności ustępuje on miejsca SQLite. Access jest jednak narzędziem często wykorzystywanym w środowiskach biznesowych, dlatego

warto je znać. Daje ono możliwość pisania zapytań bez użycia języka SQL dzięki przystosowanym do tego celu narzędziom wizualnym, a także projektowania bazy danych za pomocą interfejsu graficznego lub tworzenia makr. Wiele firm poszukuje osób do obsługi baz danych Microsoft Access oraz ich utrzymywania lub też wykonywania migracji danych do bardziej pożądaných platform bazodanowych, takich jak MySQL.

Scentralizowane bazy danych

Jeśli z bazy danych korzystać będą jednocześnie dziesiątki, setki, a nawet tysiące użytkowników i aplikacji, lekkie bazy danych nie poradzą sobie z taką ilością zapytań. Potrzebna będzie scentralizowana baza danych, która wymaga serwera i jest w stanie sprawnie obsłużyć dużą liczbę poleceń. Obecnie na rynku jest dostępnych wiele różnego typu rozwiązań scentralizowanych baz danych, np.:

- MySQL,
- Microsoft SQL Server,
- Oracle,
- PostgreSQL,
- Teradata,
- IBM DB2,
- MariaDB.

Niektóre z tych rozwiązań możesz zainstalować na każdego rodzaju komputerze, aby następnie przekształcić go w serwer. Następnie możesz podłączyć do niego komputery użytkowników, tzw. **klienty** (ang. *clients*), aby mogli oni korzystać z zamieszczonej na nim bazy danych. Klient może wysłać zapytanie SQL, żądając konkretnych danych. Zadaniem serwera jest przetworzenie tego żądania i zwrócenie odpowiedzi. Jest to klasyczna **konfiguracja klient-serwer**. Klient wysłał żądanie pewnych danych, a serwer w odpowiedzi je zwraca.

Istnieje wprowadzie możliwość przekształcenia w serwer bazy danych MySQL dowolnego komputera, takiego jak MacBook lub tani pecet, jednak większe natężenie ruchu w wymianie danych pomiędzy klientami a serwerem wymaga zastosowania bardziej specjalistycznych komputerów, określaných mianem **komputerów-serwerów** (ang. *server computers*). Są one przystosowane do wykonywania zadań dedykowanych właśnie dla serwerów. Tego typu urządzenia są zazwyczaj utrzymywane przez dział IT. Jego pracownicy zajmują się administrowaniem

bazami danych, których prawidłowe działanie zostało uznane za kluczowe dla funkcjonowania przedsiębiorstwa, oraz je nadzorują.



Czasami terminu SQL używa się błędnie w odniesieniu do nazw platform bazodanowych, takich jak MySQL, Microsoft SQL Server lub SQLite, podczas gdy SQL jest uniwersalnym językiem służącym do pracy z danymi umieszczonymi na tych platformach. Słowo SQL zostało użyte w ich nazwach jedynie dla celów marketingowych.

Jako nowy pracownik będziesz musiał poprosić o dostęp do potrzebnych Ci informacji, znajdujących się w scentralizowanej bazie danych, która prawdopodobnie będzie istnieć w firmie, do której dołączasz. Chociaż w tej książce nie będziemy poruszać tematów związanych z tego rodzaju bazą danych, to jednak omawiane w niej tematy w dużej mierze pokrywają się z zagadnieniami dotyczącymi scentralizowanej bazy danych. W przypadku wszystkich rozwiązań bazodanowych do wyświetlania wybranych informacji z tabel język SQL wykorzystywany jest w analogiczny sposób. Także edytory, za pomocą których możliwe jest odpytywanie bazy danych za pomocą tego języka, różnią się od siebie tylko w niewielkim stopniu. Zdarza się jednak, że w poszczególnych rozwiązaniach dostępnych na rynku składnia niektórych instrukcji w języku SQL nie jest taka sama, jak na przykład ma to miejsce w odniesieniu do funkcji związanych z datami. Należy jednak zaznaczyć, że elementy tego języka zaprezentowane w niniejszej publikacji należy traktować jako powszechnie stosowane.

Jeśli więc kiedykolwiek staniesz przed wyzwaniem zbudowania scentralizowanej bazy danych, z czystym sumieniem poleciłbym Ci MySQL. Jest to rozwiązanie typu *open source*, z którego można swobodnie korzystać, a jego instalacja i konfiguracja są bardzo proste. Korzysta z niego Facebook, Google, eBay, Twitter i setki innych firm z Doliny Krzemowej.

Teraz, gdy poznałeś już podstawy organizacji baz danych, możemy przejść do nauki korzystania z ich dobrodziejstw. W tej książce będziemy posługiwać się systemem zarządzania bazą danych SQLite. Rodzaj systemu nie ma jednak większego znaczenia, gdyż zastosowanie języka SQL w przypadku innych tego rodzaju rozwiązań jest analogiczne. Dlatego też wszystko to, czego się dowiesz, czytając tę książkę, będziesz mógł zastosować do wszystkich innych platform bazodanowych.

!=, 126
%, 44, 46, 127
_, 46, 127
||, 39, 40, 127
<, 126
<=, 43, 126
<>, 43, 126
=, 45, 126
==, 126
>, 126
>=, 43, 126

A

aggregating data, *Patrz:*
 dane agregowanie
alias, 36, 37
analityka biznesowa, 121,
 123
atomowość, 136

B

baza danych, 17
 administrator, 88
 architektura, 87
 dodawanie, 25
 lekka, 20
 odpytywanie, 22
 relacyjna, 17, 20

 scentralizowana, 20, 21
 schemat, 94
 struktura, 27, 94
 tworzenie, 95, 97
 zasilanie, 88
business intelligence,
 Patrz: analityka
biznesowa

D

dane
 agregowanie, 53, 57, 65
 baza, *Patrz:* baza
 danych
 bezpieczeństwo, 88, 89
 wstrzykiwanie kodu
 SQL, 89
 eksploracja, 123
 migracja, 20
 modyfikowanie, 113,
 115
 ograniczenie
 AUTOINCREMENT,
 103, 104, 111, 115
 DEFAULT, 111
 FOREIGN KEY, 111
 NOT NULL, 111,
 115

 PRIMARY KEY, 111
 usuwanie, 118
 typ, *Patrz:* typ
 zarządzanie, 113
 zwijanie, *Patrz:* dane
 agregowanie
 źródłowe, 88
data mining, *Patrz:* dane
eksploracja
database administrator,
 Patrz: baza danych
 administrator
DBA, *Patrz:* baza danych
 administrator

F

foreign key, *Patrz:* klucz
 obcy
format
 CSV, 17
 ISO8601, 130, 131
 XML, 17
funkcja
 ABS, 128
 agregująca, 56, 58, 129
 AVG, 56, 57, 129
 COALESCE, 50, 128
 CONCAT, 40

funkcja

COUNT, 53, 56, 129
DATE, 131
DATETIME, 132
daty i czasu, 130, 131, 132
GROUP_CONCAT, 129
INSTR, 46, 47, 128
LENGHT, 128
LIKE, 47
LOWER, 129
LTRIM, 128
MAX, 56, 57, 67, 129
MIN, 56, 57, 67, 129
RANDOM, 128
REPLACE, 46, 129
ROUND, 38, 57, 129
RTRIM, 128
SUBSTR, 46, 47, 129
SUM, 56, 57, 58, 60, 129
tekstowa, 47
TIME, 132
TRIM, 128
UPPER, 129
wbudowana, 38, 128
znakowa, 46

H

Hipp Richard, 24

I

IBM DB2, 21, 43
indeks, 111, 133, 134
tworzenie, 135
typu UNIQUE, 136
usuwanie, 136
instrukcja
CASE, 63, 64, 66
grupowanie, 65

NULL, 65

wartość 0, 65

CREATE TABLE, 97,
101, 102, 104

CREATE VIEW, 110

DELETE, 113, 117,
118, 133, 134, 136

DROP INDEX, 136

INSERT, 113, 114, 133,
134, 136, 137

wstawianie kilku

wierszy, 116

SELECT, 31, 32, 34,
41, 64, 67, 83, 123
wydajność, 111, 134

TRUNCATE TABLE,
118

UPDATE, 113, 119,
133, 134, 136

interfejs użytkownika, 25

J

język

C#, 123

Java, 113

Python, 113, 123

R, 123

Swift, 123

K

klauszula

GROUP BY, 34, 54,
55, 83, 129

HAVING, 59

INNER JOIN, 74, 75,
76, 85

IS NOT NULL, 48

IS NULL, 48

JOIN, 83

LEFT JOIN, 78, 79, 80,
85

OUTER JOIN, 80, 81

RIGHT JOIN, 80, 81

WHERE, 41, 42, 49,
50, 59, 66, 118, 119

klient, 21

klient-serwer, 21

klucz

główny, 92, 93, 94,
103, 104

obcy, 93, 94, 107
definiowanie, 106
ograniczenia, 108
weryfikacja, 117

komputer-serwer, 21

konfiguracja klient-
serwer, *Patrz:* klient-
serwer

konkatenacja, 39, 40, 127

L

liczba pseudolosowa, 128
literał, 45, 125

M

MariaDB, 21

Microsoft Access, 20, 43

Microsoft SQL Server, 21,
22

MySQL, 21, 22, 104, 115,
118

N

normalizacja, 18

NULL, 48, 49, 57, 65, 67,
85

O

obszar roboczy, 31
operator
 !=, 126
 %, 44, 46
 ||, 127
 <, 126
 <=, 43, 126
 <>, 43, 126
 =, 45, 126
 ==, 126
 >, 126
 >=, 43, 126
AND, 43, 45, 50, 127
BETWEEN, 43, 127
DISTINCT, 61
IN, 44, 45, 127
IS NOT NULL, 127
IS NULL, 127
JOIN, 71
konkatenacji, 39
LIKE, 46, 127, 134
logiczny, 127
matematyczny, 38, 126
modulo, 44, 46
NOT, 127
NOT IN, 44
OR, 44, 45, 50, 127
porównania, 126
REGEXP, 128
tekstowy, 127
UNION, 133
UNION ALL, 133
Oracle, 21

P

podzapytanie, 133
PostgreSQL, 21
primary key, *Patrz:* klucz
 główny

R

RDBMS, *Patrz:* system
 zarządzania relacyjną
 bazą danych
relacja, 72, 92, 94
 jeden do jednego, 73
 jeden do wielu, 73, 93
Relational Database
 Management System,
 Patrz: system
 zarządzania relacyjną
 bazą danych

S

server computer, *Patrz:*
 komputer-serwer
serwer, 20, 21
słowo kluczowe
 AND, 68
 AS, 36
 CASE, 63
 CREATE, 87
 DISTINCT, 61
 ELSE, 63
 END, 63
 false, 47
 FROM, 42, 75
 HAVING, 60
 INNER JOIN, 85
 JOIN, 74, 83
 LEFT JOIN, 85
 NOT, 68
 OR, 68
 SELECT, 42
 SET, 119
 true, 47
 WHERE, 42
SQL, 22

SQL injection, *Patrz:* dane
 bezpieczeństwo
 wstrzykiwanie kodu SQL
SQLite, 20, 22, 23, 90, 106,
 108
SQLiteStudio, 24, 95, 98,
 135
Stack Overflow, 15
system zarządzania
 relacyjną bazą danych, 17,
 23

T

tabela, 28
 dodawanie, 87
 kolumna, 34
 definicja, 103
 liczba porządkowa,
 55
 nazwa, 37, 42, 93
 nadrzędna, 72, 81, 93
 nazwa, 37, 42
 podrzędna, 72, 80, 81,
 93, 107
 relacja, *Patrz:* relacja
 tworzenie, 97, 98, 101,
 102, 104
wiersz, 41
 filtrowanie, 42
 grupowanie, 53
 liczenie, 53
 sortowanie, 56
 unikalny, 61
 usuwanie, 117
 złączenie, 75, 81, 92
Teradata, 21
transakcja, 133, 136, 137
typ, 29
 boolean, 47
 tekstowy, 39, 40, 106

V

view, 108, *Patrz:* widok

W

wartość

NULL, *Patrz:* NULL

tekstowa, 106

pusta, 49

widok, 108, 110

wildcard, *Patrz:* znak

wieloznaczny

wyrażenie

CASE, 63

regularne, 47, 128, 134

wyzwalacz, 134

wzorzec tekstowy, 47, 134

Z

zapytanie, 21, 28, 32, 108

struktura, 36

znak

%, 44, 46, 127

_, 46, 127

||, 39, 40

podkreślenia, 37

pojedynczego

cudzysłowu, 45

średnik, 34

wieloznaczny, 46, 127,

134

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ 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 WYDAWNICZA

 **Helion SA**

Dzisiejsze firmy gromadzą ogromne ilości danych. Dane te jednak mogą przynieść wymierną korzyść jedynie wtedy, gdy są w odpowiedni sposób przechowywane oraz należycie zabezpieczone i jeśli umie się z nich korzystać. Bywa, że przeszukiwanie dużych zbiorów danych i odnajdywanie w nich potrzebnych informacji czy ich przetworzenie staje się nie lada wyzwaniem. Kiedy pocziwy arkusz Excela staje się niewystarczający, doskonale sprawdzają się relacyjne bazy danych, które najczęściej można obsłużyć za pomocą instrukcji pisanych w języku SQL.

Wielu osobom nauka języka SQL może przysparzać problemów. Trzymasz w ręku świetnie napisany podręcznik, który ułatwi Ci to zadanie! Przewodnik przedstawia praktyczne sposoby wykorzystania języka SQL bez zagłębiania się w techniczne szczegóły jego działania. Najwięcej miejsca poświęcono na ćwiczenia z rzeczywistymi bazami danych. Książka ta pomaga w uzyskaniu praktycznej wiedzy nie tylko o pracy z bazami danych, ale także o ich wykorzystaniu do rozwiązywania problemów biznesowych.

Thomas Nield – programista i analityk z wieloletnim doświadczeniem. Obecnie zajmuje się rozwijaniem oprogramowania dla biznesu w firmie Southwest Airlines. Specjalista w zakresie programowania reaktywnego, programowania w językach Java i Kotlin, a także oprogramowania dla biznesu do realizacji zadań strategicznych. Autor wielu popularnych artykułów i uczestnik licznych projektów programistycznych.

Najważniejsze zagadnienia omówione w książce:

- podstawy wiedzy o relacyjnych bazach danych i ich rodzajach oraz o języku SQL
- baza danych SQLite
- polecenie SELECT, ważniejsze klauzule i słowa kluczowe
- podstawy projektowania baz danych
- operatory i funkcje języka SQL

Helion

księgarnia Internetowa

<http://helion.pl>

zamówienia telefoniczne



0 801 339900



0 601 339900

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
<http://helion.pl>

Sprawdź najnowsze promocje:
● <http://helion.pl/promocje>
Książki najchętniej czytane:
● <http://helion.pl/bestsellery>
Zamów informacje o nowościach:
● <http://helion.pl/nowości>

sięgnij po **WIĘCEJ**



KOD KORZYŚCI

ISBN 978-83-283-2818-1



9 788328 328181

Informatyka w najlepszym wydaniu

cena: 34,90 zł