PHP i MySQL. Tworzenie stron WWW. Wydanie drugie. Vademecum profesjonalisty

Autorzy: Luke Welling, Laura Thomson
Tłumaczenie: Daniel Kaczmarek, Łukasz Zieliński
ISBN: 83-7361-140-1
Tytuł oryginału: PHP and MySQL Web Development, Second Edition
Format: B5, stron: 840

PHP i MySQL to wydajny tandem, pozwalający na realizację wielu projektów związanych z programowaniem aplikacji internetowych. Autorzy książki „PHP i MySQL. Tworzenie stron WWW. Wydanie drugie. Vademecum profesjonalisty" stworzyli więc unikatowy podręcznik, który łączy w sobie opis funkcjonalności PHP i MySQL z prezentacją wielu praktycznych rozwiązań, stworzonych za pomocą tych narzędzi. To podejście okazało się trafione, czego dowodem jest ogromna popularność pierwszego wydania tej książki.

Oprócz składni i biblioteki funkcji PHP, książka opisuje również podstawowe pojęcia z dziedziny profesjonalnej inżynierii oprogramowania związanego z siecią WWW. Niektóre, takie jak utrzymywanie, współpraca i testowanie, są kwestiami istotnymi dla inżynierów oprogramowania. Inne pojęcia, takie jak uwierzytelnianie, szyfrowanie i kontrola sesji, mają szczególne znaczenie dla projektów programistycznych opartych na Internecie.

- Podstawowy kurs PHP
- Tworzenie i obsługa baz danych za pomocą MySQL
- Dynamiczne tworzenie kodu HTML, obrazków i dokumentów
- Tworzenie bezpiecznych usług za pomocą uwierzytelniania i SSL
- Tworzenie koszyka na zakupy dla witryn handlu elektronicznego
- Opis praktyk związanych z inżynierią oprogramowania dla większych projektów WWW
- Zastosowanie obiektowych technik programistycznych
- Tworzenie spersonalizowanych dokumentów PDF
- Korzystanie z usług WWW za pomocą XML i SOAP

„PHP i MySQL. Tworzenie stron WWW. Wydanie drugie. Vademecum profesjonalisty" to książka, która nie tylko przekazuje wiedzę na temat PHP i MySQL, ale także prezentuje praktyczne sposoby jej wykorzystania. Jeśli chcesz szybko nauczyć się tworzenia profesjonalnych aplikacji WWW, książka ta będzie trafionym zakupem.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

O Autorach ... 19

Wprowadzenie ... 21

Część I Stosowanie PHP ... 29

Rozdział 1. Podstawowy kurs PHP ... 31

Zastosowanie PHP ... 32
Przykładowa aplikacja: „Części samochodowe Janka” ... 32
Formularz zamówienia ... 32
Przetwarzanie formularza ... 34
Osadzanie PHP w HTML ... 34
Zastosowanie znaczników PHP ... 35
Style znaczników PHP ... 36
Instrukcje PHP ... 36
Odstępy ... 37
Komentarze ... 37
Dodawanie zawartości dynamicznej ... 38
Wywoływanie funkcji ... 39
Funkcja date() ... 39
Dostęp do zmiennych formularza ... 39
Zmienne formularza ... 40
Łączenie ciągów ... 42
Zmienne i ciągi znaków ... 43
Identyfikatory ... 44
Zmienne zadeklarowane przez użytkownika ... 44
Przypisywanie wartości zmiennym ... 44
Typy zmiennych ... 45
Typy danych w PHP ... 45
Siła typu ... 45
Rzutowanie typu ... 46
Zmienne zmiennych ... 46
Stałe ... 47
Zasięg zmiennych ... 47
Operatory ... 48
Operatory arytmetyczne ... 49
Operatory ciągów ... 50
Operatory przypisania ... 50
Operatory porównań ... 52
Operatory logiczne ... 53

Operatory bitowe ... 54
Pozostałe operatory ... 54
Stosowanie operatorów: obliczanie sum w formularzu ... 56
Pierwszeństwo i kolejność: wyznaczanie wartości wyrażeń ... 57
Zarządzanie zmiennymi ... 58
Sprawdzanie i ustawianie typów zmiennych ... 58
Sprawdzanie stanu zmiennej ... 59
Reinterpretacja zmiennych ... 60
Struktury kontrolujące ... 60
Podejmowanie decyzji za pomocą instrukcji warunkowych ... 60
Instrukcja if ... 61
Bloki kodu ... 61
Instrukcja else ... 62
Instrukcja elseif ... 62
Instrukcja switch ... 63
Porównanie różnych instrukcji warunkowych ... 65
Iteracja: powtarzanie działań ... 65
Pętle while ... 66
Pętle for i foreach ... 67
Pętle do..while ... 69
Wyłamywanie się ze struktury skryptu ... 69
W następnym rozdziale: zapisywanie zamówienia klienta ... 69

Rozdział 2. Przechowywanie i wyszukiwanie danych ... 71
Zapisywanie danych do późniejszego użycia ... 71
Przechowywanie i wyszukiwanie zamówień Janka ... 72
Podstawowe informacje na temat przetwarzania plików ... 73
Otwieranie pliku ... 73
Tryby otwarcia pliku ... 73
Stosowanie funkcji fopen() do otwarcia pliku ... 74
Otwieranie pliku przez protokół FTP lub HTTP ... 76
Problemy z otwieraniem plików ... 77
Zapisywanie danych w pliku ... 79
Parametry funkcji fwrite() ... 79
Formaty plików ... 79
Zamykanie pliku ... 80
Odczyt z pliku ... 80
Otwieranie pliku w celu odczytu — fopen() ... 81
Wiedzieć, kiedy przestać — feof() ... 82
Odczytywanie pliku wiersz po wierszu — fgets(), fgetss() i fgetcsv() ... 82
Odczyt całego pliku — readfile(), fpassthru(), file() ... 83
Odczyt pojedynczego znaku — fgetc() ... 84
Odczytywanie zadanej długości — fread() ... 84
Inne przydatne funkcje plikowe ... 85
Sprawdzanie istnienia pliku — file_exists() ... 85
Określanie wielkości pliku — filesize() ... 85
Kasowanie pliku — unlink() ... 85
Poruszanie się wewnątrz pliku — rewind(), fseek() i ftell() ... 85
Blokowanie pliku ... 86
Lepszy sposób obróbki danych — systemy zarządzania bazami danych ... 88
Problemy związane ze stosowaniem plików jednorodnych ... 88
Jak RDBMS rozwiązują powyższe problemy? ... 88
Propozycje dalszych lektur ... 89
W następnym rozdziale ... 89

Rozdział 3. Stosowanie tablic ... 91
Czym są tablice? ... 91
Tablice indeksowane numerycznie ... 92
Inicjowanie tablic indeksowanych numerycznie ... 92
Dostęp do zawartości tablicy ... 93
Dostęp do tablic przy zastosowaniu pętli ... 94
Tablice asocjacyjne ... 94
Inicjowanie tablicy asocjacyjnej ... 94
Dostęp do elementów tablicy ... 95
Stosowanie pętli z tablicami asocjacyjnymi ... 95
Tablice wielowymiarowe ... 97
Sortowanie tablic ... 100
Stosowanie funkcji sort() ... 100
Stosowanie funkcji asort() i ksort() do porządkowania tablic asocjacyjnych ... 101
Sortowanie odwrotne ... 101
Sortowanie tablic wielowymiarowych ... 101
Typy sortowań definiowane przez użytkownika ... 102
Odwrotne sortowanie zdefiniowane przez użytkownika ... 103
Zmiany kolejności elementów w tablicach ... 104
Stosowanie funkcji shuffle() ... 104
Stosowanie funkcji array_reverse() ... 105
Ładowanie tablic z plików ... 106
Inne działania na tablicach ... 109
Poruszanie się wewnątrz tablicy — funkcje each(), current(), reset(), end(), next(), pos() i prev() ... 109
Dołączanie dowolnej funkcji do każdego elementu tablicy — funkcja array_walk() ... 110
Liczenie elementów tablicy: count(), sizeof() i array_count_values() ... 111
Konwersja tablic na zmienne skalarne — funkcja extract() ... 112
Propozycje dalszych lektur ... 113
W następnym rozdziale ... 113

Rozdział 4. Manipulowanie ciągami i wyrażenia regularne ... 115
Przykładowa aplikacja — Inteligentny Formularz Pocztowy ... 115
Formatowanie ciągów ... 117
Przycinanie ciągów — funkcje chop(), ltrim() i trim() ... 117
Formatowanie ciągów w celu ich prezentacji ... 118
Formatowanie ciągów do przechowania — funkcje AddSlashes() i StripSlashes() ... 121
Łączenie i rozdzielanie ciągów za pomocą funkcji ciągów ... 122
Stosowanie funkcji explode(), implode() i join() ... 123
Stosowanie funkcji strtok() ... 124
Stosowanie funkcji substr() ... 124
Porównywanie ciągów ... 125
Porządkowanie ciągów — funkcje strcmp(), strcasecmp() i strnatcmp() ... 125
Sprawdzanie długości ciągu za pomocą funkcji strlen() ... 126
Dopasowywanie i zamiana podciągów za pomocą funkcji ciągów ... 126
Znajdowanie ciągów w ciągach — funkcje strstr(), strchr(), strrchr() i stristr() ... 127
Odnajdywanie pozycji podciągu — funkcje strpos() i strrpos() ... 128
Zamiana podciągów — funkcje str_replace() i substr_replace() ... 129
Wprowadzenie do wyrażeń regularnych ... 130
Podstawy ... 130
Zbiory i klasy znaków ... 131
Powtarzalność ... 132
Podwyrażenia ... 132
Podwyrażenia policzalne ... 132
Kotwiczenie na początku lub na końcu ciągu ... 133
Rozgałęzianie ... 133

Dopasowywanie specjalnych znaków literowych ... 133
Podsumowanie znaków specjalnych ... 133
Umieszczanie wszystkiego razem (Inteligentny Formularz) ... 134
Odnajdywanie podciągów za pomocą wyrażeń regularnych ... 135
Zamiana podciągów za pomocą wyrażeń regularnych ... 136
Rozdzielanie ciągów za pomocą wyrażeń regularnych ... 136
Porównanie funkcji ciągów i funkcji wyrażeń regularnych ... 137
Propozycje dalszych lektur ... 137
W następnym rozdziale ... 137

Rozdział 5. Ponowne wykorzystanie kodu i tworzenie funkcji ... 139
Dlaczego ponownie stosować kod? ... 139
Koszt ... 140
Niezawodność ... 140
Spójność ... 140
Stosowanie funkcji require() i include() ... 140
Stosowanie funkcji require() ... 141
Rozszerzenia plików i require() ... 142
Znaczniki PHP i require() ... 142
Stosowanie require() w szablonach stron WWW ... 142
Stosowanie opcji auto_prepend_file i auto_append_file ... 147
Stosowanie funkcji include() ... 148
Stosowanie funkcji w PHP ... 149
Wywoływanie funkcji ... 150
Wywołanie niezdefiniowanej funkcji ... 151
Wielkość liter a nazwy funkcji ... 152
Dlaczego powinno się definiować własne funkcje? ... 152
Podstawowa struktura funkcji ... 152
Nadawanie nazwy funkcji ... 153
Parametry ... 154
Zasięg ... 156
Przekazanie przez referencję czy przekazanie przez wartość? ... 158
Powrót z funkcji ... 159
Zwracanie wartości przez funkcje ... 160
Bloki kodu ... 161
Rekurencja ... 162
Propozycje dalszych lektur ... 164
W następnym rozdziale ... 164

Rozdział 6. Obiektowy PHP ... 165
Koncepcje programowania obiektowego ... 165
Klasy i obiekty ... 165
Polimorfizm ... 167
Dziedziczenie ... 167
Tworzenie klas, atrybutów i operacji w PHP ... 168
Struktura klasy ... 168
Konstruktory ... 168
Tworzenie egzemplarzy ... 169
Stosowanie atrybutów klasy ... 169
Wywoływanie operacji klas ... 171
Implementacja dziedziczenia w PHP ... 172
Unieważnianie ... 173
Wielodziedziczenie ... 174
Tworzenie klas ... 175
Tworzenie kodu dla własnej klasy ... 175
W następnej części ... 183

Część II Stosowanie MySQL ... 185

Rozdział 7. Projektowanie internetowej bazy danych ... 187
Koncepcje relacyjnych baz danych ... 188
Tabele ... 188
Kolumny ... 188
Wiersze ... 188
Wartości ... 188
Klucze ... 189
Schematy ... 190
Relacje ... 190
Jak zaprojektować internetową bazę danych? ... 191
Określ obiekty świata realnego, których model chcesz wykonać ... 191
Unikaj przechowywania redundantnych danych ... 191
Zapisuj atomowe wartości kolumn ... 193
Dobierz właściwe klucze ... 194
Pomyśl o zapytaniach, które zadasz bazie ... 194
Unikaj tworzenia tabel z wieloma pustymi polami ... 194
Typy tabel — podsumowanie ... 195
Architektura internetowej bazy danych ... 196
Architektura ... 196
Propozycje dalszych lektur ... 197
W następnym rozdziale ... 197

Rozdział 8. Tworzenie internetowej bazy danych ... 199
Uwagi na temat użytkowania monitora MySQL ... 200
Jak zalogować się do serwera MySQL ... 201
Tworzenie baz i rejestrowanie użytkowników ... 202
Tworzenie bazy danych ... 202
Użytkownicy i przywileje ... 203
Wprowadzenie do systemu przywilejów MySQL ... 203
Zasada najmniejszego przywileju ... 203
Rejestrowanie użytkowników: polecenie GRANT ... 204
Typy i poziomy przywilejów ... 205
Polecenie REVOKE ... 207
Przykłady użycia poleceń GRANT i REVOKE ... 207
Rejestrowanie użytkownika łączącego się z Internetu ... 208
Wylogowanie się użytkownika root ... 209
Używanie odpowiedniej bazy danych ... 209
Tworzenie tabel bazy danych ... 210
Znaczenie dodatkowych atrybutów kolumn ... 211
Typy kolumn ... 212
Rzut oka na bazę danych — polecenia SHOW i DESCRIBE ... 214
Identyfikatory MySQL ... 215
Typy danych w kolumnach ... 216
Typy liczbowe ... 216
Propozycje dalszych lektur ... 220
W następnym rozdziale ... 220

Rozdział 9. Praca z bazą danych MySQL ... 221
Czym jest SQL? ... 221
Zapisywanie danych do bazy ... 222
Wyszukiwanie danych w bazie ... 224
Wyszukiwanie danych spełniających określone kryteria ... 225
Wyszukiwanie danych w wielu tabelach ... 227

Szeregowanie danych w określonym porządku ... 232
Grupowanie i agregowanie danych ... 233
Wskazanie wierszy, które mają być wyświetlone ... 235
Dokonywanie zmian rekordów w bazie danych ... 235
Zmiana struktury istniejących tabel ... 236
Usuwanie rekordów z bazy danych ... 237
Usuwanie tabel ... 238
Usuwanie całych baz danych ... 238
Propozycje dalszych lektur ... 238
W następnym rozdziale ... 238

Rozdział 10. Łączenie się z bazą MySQL za pomocą PHP ... 239
Jak działa internetowa baza danych? ... 240
Etapy wysyłania zapytań do bazy danych z poziomu strony WWW ... 243
Sprawdzenie poprawności wpisanych danych ... 243
Ustanawianie połączenia z bazą danych ... 244
Wybór właściwej bazy danych ... 246
Wysyłanie zapytań do bazy danych ... 246
Odczytywanie rezultatów zapytań ... 247
Zamykanie połączenia z bazą danych ... 248
Wstawianie nowych danych do bazy ... 248
Inne użyteczne funkcje PHP i MySQL ... 252
Zwalnianie zasobów ... 252
Tworzenie i usuwanie baz danych ... 252
Inne interfejsy bazodanowe PHP ... 252
Stosowanie ogólnego intrefejsu bazodanowego: PEAR DB ... 253
Propozycje dalszych lektur ... 256
W następnym rozdziale ... 256

Rozdział 11. MySQL dla zaawansowanych ... 257
Szczegóły systemu przywilejów ... 257
Tabela user ... 258
Tabele db i host ... 259
Tabele tables_priv i columns_priv ... 261
Kontrola dostępu: w jaki sposób MySQL używa tabel przywilejów ... 261
Zmiana przywilejów: kiedy zmiany zostaną uwzględnione? ... 262
Ochrona bazy danych ... 263
MySQL z perspektywy systemu operacyjnego ... 263
Hasła ... 263
Przywileje użytkowników ... 264
MySQL i Internet ... 265
Uzyskiwanie szczegółowych informacji o bazie danych ... 265
Uzyskiwanie informacji poleceniem SHOW ... 266
Uzyskiwanie informacji o kolumnach za pomocą polecenia DESCRIBE ... 267
Jak wykonywane są zapytania: polecenie EXPLAIN ... 268
Przyspieszanie wykonania zapytań za pomocą indeksów ... 271
Wskazówki dotyczące optymalizacji ... 271
Optymalizacja projektu bazy danych ... 272
Przywileje ... 272
Optymalizacja tabel ... 272
Stosowanie indeksów ... 272
Używanie wartości domyślnych ... 273
Używanie stałych połączeń z bazą ... 273
Więcej wskazówek ... 273
Różne typy tabel ... 273

Ładowanie danych z pliku ... 274
Tworzenie kopii zapasowej bazy danych MySQL ... 274
Przywracanie bazy danych MySQL ... 275
Propozycje dalszych lektur ... 275
W następnej części ... 275

Część III E-commerce i bezpieczeństwo ... 277

Rozdział 12. Komercyjne witryny internetowe ... 279
Co chcemy osiągnąć? ... 279
Rodzaje komercyjnych stron WWW ... 279
Broszury internetowe ... 280
Przyjmowanie zamówień na produkty i usługi ... 283
Dostarczanie usług lub wyrobów mających postać cyfrową ... 287
Zwiększanie wartości produktów i usług ... 288
Ograniczanie kosztów ... 288
Ryzyko i zagrożenia ... 289
Crackerzy ... 289
Przyciągnięcie niewystarczającej liczby klientów ... 290
Awarie sprzętu komputerowego ... 290
Awarie sieci elektrycznych, komunikacyjnych i komputerowych oraz systemu wysyłkowego ... 291
Silna konkurencja ... 291
Błędy w oprogramowaniu ... 291
Zmiany polityki rządowej ... 292
Ograniczenie pojemności systemów ... 292
Wybór strategii ... 292
W następnym rozdziale ... 293

Rozdział 13. Bezpieczeństwo komercyjnych stron WWW ... 295
Jaką wagę mają posiadane przez nas informacje? ... 296
Zagrożenia bezpieczeństwa ... 296
Ujawnienie informacji poufnych ... 297
Utrata lub zniszczenie danych ... 299
Modyfikacje danych ... 299
Blokada usługi ... 300
Błędy w oprogramowaniu ... 301
Zaprzeczenie korzystania z usługi ... 303
Równoważenie użyteczności, wydajności, kosztów i bezpieczeństwa ... 304
Opracowanie polityki bezpieczeństwa ... 305
Zasady uwierzytelniania ... 305
Wykorzystanie mechanizmu uwierzytelniania ... 306
Podstawy szyfrowania ... 307
Szyfrowanie z kluczem prywatnym ... 308
Szyfrowanie z kluczem publicznym ... 309
Podpis cyfrowy ... 310
Certyfikaty cyfrowe ... 311
Bezpieczne serwery WWW ... 312
Monitorowanie i zapisywanie zdarzeń ... 314
Zapory sieciowe ... 314
Tworzenie kopii zapasowych ... 315
Tworzenie kopii zapasowych zwykłych plików ... 315
Tworzenie kopii zapasowych i odzyskiwanie baz danych MySQL ... 316
Bezpieczeństwo fizyczne ... 316
W następnym rozdziale ... 317

Rozdział 14. Uwierzytelnianie przy użyciu PHP i MySQL ... 319
Identyfikacja użytkowników ... 319
Implementacja kontroli dostępu ... 320
Przechowywanie haseł dostępu ... 323
Szyfrowanie haseł ... 325
Zastrzeganie więcej niż jednej strony ... 327
Podstawowa metoda uwierzytelniania ... 327
Wykorzystanie podstawowej metody uwierzytelniania w PHP ... 329
Wykorzystanie podstawowej metody uwierzytelniania na serwerze Apache przy użyciu plików .htaccess ... 330
Wykorzystanie podstawowej metody uwierzytelniania na serwerze IIS ... 334
Wykorzystanie modułu mod_auth_mysql do celów uwierzytelniania ... 337
Instalacja modułu mod_auth_mysql ... 337
Zadziałało? ... 338
Praca z mod_auth_mysql ... 338
Implementacja własnej metody uwierzytelniania ... 339
Propozycje dalszych lektur ... 340
W następnym rozdziale ... 340

Rozdział 15. Zabezpieczanie transakcji przy użyciu PHP i MySQL ... 341
Zapewnienie bezpieczeństwa transakcji ... 341
Komputer użytkownika ... 342
Internet ... 344
System docelowy ... 345
Wykorzystanie protokołu Secure Sockets Layer (SSL) ... 346
Kontrola danych pochodzących od użytkownika ... 349
Bezpieczne przechowywanie danych ... 350
Cel przechowywania numerów kart kredytowych ... 352
Szyfrowanie danych w PHP ... 352
Propozycje dalszych lektur ... 361
W następnej części ... 361

Część IV Zaawansowane techniki PHP ... 363

Rozdział 16. Interakcja z systemem plików i serwerem ... 365
Wprowadzenie do wysyłania plików ... 365
Kod HTML służący do wysyłania plików ... 366
Uwaga na temat bezpieczeństwa ... 367
Tworzenie obsługującego plik PHP ... 367
Najczęściej spotykane problemy ... 372
Stosowanie funkcji katalogowych ... 372
Odczyt z katalogów ... 373
Otrzymywanie informacji na temat aktualnego katalogu ... 374
Tworzenie i usuwanie katalogów ... 374
Interakcja z systemem plików ... 375
Otrzymywanie informacji o pliku ... 375
Zmiana właściwości pliku ... 378
Tworzenie, usuwanie i przenoszenie plików ... 378
Stosowanie funkcji uruchamiających programy ... 379
Interakcja ze środowiskiem: funkcje getenv() i putenv() ... 382
Propozycje dalszych lektur ... 382
W następnym rozdziale ... 382

Rozdział 17. Stosowanie funkcji sieci i protokołu ... 383
Przegląd protokołów ... 383
Wysyłanie i odczytywanie poczty elektronicznej ... 384

Korzystanie z innych usług WWW ... 384
Stosowanie funkcji połączeń sieciowych ... 387
Korzystanie z FTP ... 391
Stosowanie FTP w celu utworzenia kopii bezpieczeństwa lub kopii lustrzanej pliku ... 391
Wysyłanie plików ... 397
Unikanie przekroczenia dopuszczalnego czasu ... 398
Stosowanie innych funkcji FTP ... 398
Stosowanie ogólnej komunikacji sieciowej za pomocą cURL ... 399
Propozycje dalszych lektur ... 401
W następnym rozdziale ... 402

Rozdział 18. Zarządzanie datą i czasem ... 403
Uzyskiwanie informacji o dacie i czasie w PHP ... 403
Stosowanie funkcji date() ... 403
Obsługa znaczników czasu Uniksa ... 405
Stosowanie funkcji getdate() ... 406
Sprawdzanie poprawności dat ... 406
Konwersja pomiędzy formatami daty PHP i MySQL ... 407
Obliczanie dat ... 407
Stosowanie funkcji kalendarzowych ... 409
Propozycje dalszych lektur ... 410
W następnym rozdziale ... 410

Rozdział 19. Generowanie obrazków ... 411
Konfigurowanie obsługi obrazków w PHP ... 411
Formaty obrazków ... 412
JPEG ... 412
PNG ... 413
WBMP ... 413
GIF ... 413
Tworzenie obrazków ... 414
Tworzenie kadru obrazka ... 415
Rysowanie lub umieszczanie tekstu w obrazku ... 416
Wyświetlanie ostatecznej grafiki ... 417
Końcowe czynności porządkujące ... 419
Stosowanie automatycznie generowanych obrazków na innych stronach ... 419
Stosowanie tekstu i czcionek do tworzenia obrazków ... 420
Konfiguracja podstawowego kadru ... 423
Dopasowanie tekstu do przycisku ... 423
Nadawanie tekstowi odpowiedniej pozycji ... 426
Wpisywanie tekstu do przycisku ... 427
Etap końcowy ... 427
Rysowanie figur i wykresów danych ... 427
Inne funkcje obrazków ... 435
Propozycje dalszych lektur ... 435
W następnym rozdziale ... 435

Rozdział 20. Stosowanie kontroli sesji w PHP ... 437
Czym jest kontrola sesji? ... 437
Podstawowa zasada działania sesji ... 438
Czym jest cookie? ... 438
Konfiguracja cookies w PHP ... 439
Stosowanie cookies w sesji ... 439
Przechowywanie identyfikatora sesji ... 440

Implementacja prostych sesji ... 440
Rozpoczynanie sesji ... 440
Zgłaszanie zmiennych sesji ... 441
Stosowanie zmiennych sesji ... 441
Usuwanie zmiennych i niszczenie sesji ... 442
Przykład prostej sesji ... 442
Konfiguracja kontroli sesji ... 445
Implementacja uwierzytelniania w kontroli sesji ... 446
Propozycje dalszych lektur ... 451
W następnym rozdziale ... 452

Rozdział 21. Inne przydatne własności ... 453
Stosowanie magicznych cudzysłowów ... 453
Wykonywanie ciągów — funkcja eval() ... 454
Zakończenie wykonania — die i exit ... 455
Serializacja ... 455
Pobieranie informacji na temat środowiska PHP ... 456
Uzyskiwanie informacji na temat załadowanych rozszerzeń ... 457
Identyfikacja właściciela skryptu ... 457
Uzyskiwanie informacji na temat daty modyfikacji skryptu ... 457
Dynamiczne dodawanie rozszerzeń ... 458
Czasowa zmiana środowiska wykonawczego ... 458
Podświetlanie źródeł ... 459
W następnej części ... 460

Część V Tworzenie praktycznych projektów PHP i MySQL ... 461

Rozdział 22. Stosowanie PHP i MySQL w dużych projektach ... 463
Zastosowanie inżynierii oprogramowania w tworzeniu aplikacji WWW ... 464
Planowanie i prowadzenie projektu aplikacji WWW ... 464
Ponowne stosowanie kodu ... 465
Tworzenie kodu łatwego w utrzymaniu ... 466
Standardy kodowania ... 466
Dzielenie kodu ... 469
Stosowanie standardowej struktury katalogów ... 470
Dokumentacja i dzielenie wewnętrznych funkcji ... 470
Implementacja kontroli wersji ... 470
Wybór środowiska programistycznego ... 472
Dokumentacja projektów ... 472
Prototypowanie ... 473
Oddzielanie logiki i zawartości ... 474
Optymalizacja kodu ... 475
Stosowanie prostych optymalizacji ... 475
Stosowanie produktów firmy Zend ... 476
Testowanie ... 476
Propozycje dalszych lektur ... 477
W następnym rozdziale ... 478

Rozdział 23. Usuwanie błędów ... 479
Błędy programistyczne ... 479
Błędy składni ... 480
Błędy wykonania ... 481
Błędy logiczne ... 486
Pomoc w usuwaniu błędów w zmiennych ... 488
Poziomy zgłaszania błędów ... 489

Zmiana ustawień zgłaszania błędów ... 491
Wyzwalanie własnych błędów ... 492
Elegancka obsługa błędów ... 492
W następnym rozdziale ... 494

Rozdział 24. Tworzenie uwierzytelniania użytkowników i personalizacji ... 495
Problem ... 495
Składniki rozwiązania ... 496
Identyfikacja użytkownika i personalizacja ... 496
Przechowywanie zakładek ... 497
Rekomendowanie zakładek ... 497
Przegląd rozwiązania ... 497
Implementacja bazy danych ... 498
Implementacja podstawowej witryny ... 500
Implementacja uwierzytelniania użytkowników ... 503
Rejestracja ... 503
Logowanie ... 509
Wylogowanie ... 512
Zmiana hasła ... 513
Ustawianie zapomnianych haseł ... 515
Implementacja przechowywania i odczytywania zakładek ... 519
Dodawanie zakładek ... 520
Wyświetlanie zakładek ... 522
Usuwanie zakładek ... 523
Implementacja rekomendacji ... 525
Rozwijanie projektu i możliwe rozszerzenia ... 529
W następnym rozdziale ... 529

Rozdział 25. Tworzenie koszyka na zakupy ... 531
Problem ... 531
Składniki rozwiązania ... 532
Tworzenie katalogu online ... 532
Śledzenie zakupów użytkownika podczas przeglądania ... 532
Płatność ... 532
Interfejs administratora ... 533
Przegląd rozwiązania ... 533
Implementacja bazy danych ... 536
Implementacja katalogu online ... 538
Przedstawianie kategorii ... 540
Wyświetlanie książek danej kategorii ... 543
Przedstawianie szczegółowych danych książki ... 544
Implementacja koszyka na zakupy ... 546
Stosowanie skryptu pokaz_kosz.php ... 546
Podgląd koszyka ... 549
Dodawanie produktów do koszyka ... 551
Zapisywanie uaktualnionego koszyka ... 553
Wyświetlanie podsumowania w pasku nagłówka ... 554
Pobyt w kasie ... 554
Implementacja płatności ... 559
Implementacja interfejsu administratora ... 561
Rozwijanie projektu ... 567
Zastosowanie istniejącego systemu ... 568
W następnym rozdziale ... 568

Rozdział 26. Tworzenie systemu zarządzania zawartością ... 569
Problem ... 569
Wymagania systemu ... 570
Edycja zawartości ... 570
Umieszczanie zawartości w systemie ... 570
Bazy danych czy pliki? ... 571
Struktura dokumentu ... 572
Stosowanie metadanych ... 572
Formatowanie danych wyjściowych ... 573
Manipulacja obrazkiem ... 574
Projekt/przegląd rozwiązania ... 576
Projektowanie bazy danych ... 576
Implementacja ... 579
Fronton systemu ... 579
Wnętrze systemu ... 582
Wyszukiwanie ... 590
Ekran redaktora naczelnego ... 593
Rozwijanie projektu ... 594
W następnym rozdziale ... 594

Rozdział 27. Tworzenie serwisu poczty elektronicznej opartego na WWW ... 595
Problem ... 595
Składniki rozwiązania ... 596
Przegląd rozwiązania ... 597
Konfiguracja bazy danych ... 599
Architektura skryptu ... 601
Logowanie i wylogowanie ... 606
Konfiguracja kont ... 609
Tworzenie nowego konta ... 611
Modyfikacja istniejącego konta ... 612
Usuwanie konta ... 612
Odczytywanie poczty ... 613
Wybór konta ... 613
Przeglądanie zawartości skrzynki ... 616
Odczytywanie wiadomości pocztowych ... 619
Przeglądanie nagłówków wiadomości ... 622
Usuwanie wiadomości ... 623
Wysyłanie wiadomości ... 624
Wysyłanie nowej wiadomości ... 624
Odpowiadanie i przekazywanie poczty ... 626
Rozwijanie projektu ... 627
W następnym rozdziale ... 628

Rozdział 28. Tworzenie menedżera list pocztowych ... 629
Problem ... 629
Składniki rozwiązania ... 630
Konfiguracja bazy danych list i abonentów ... 630
Wysyłanie plików ... 630
Wysyłanie wiadomości z załącznikami ... 631
Przegląd rozwiązania ... 631
Konfiguracja bazy danych ... 632
Architektura skryptu ... 635
Implementacja logowania ... 642
Tworzenie nowego konta ... 643
Logowanie ... 646

Implementacja funkcji użytkownika ... 648
Przeglądanie list ... 648
Przeglądanie informacji na temat listy ... 653
Przeglądanie archiwum listy ... 655
Zapisywanie i wypisywanie ... 656
Zmiana konfiguracji konta ... 657
Zmiana hasła ... 657
Wylogowanie ... 659
Implementacja funkcji administratora ... 660
Tworzenie nowej listy ... 660
Wysyłanie nowych wiadomości ... 662
Obsługa wysyłania wielu plików ... 665
Podgląd wiadomości ... 668
Rozsyłanie wiadomości ... 669
Rozwijanie projektu ... 675
W następnym rozdziale ... 676

Rozdział 29. Tworzenie forum WWW ... 677
Problem ... 677
Składniki rozwiązania ... 678
Przegląd rozwiązania ... 679
Projektowanie bazy danych ... 680
Przeglądanie drzewa artykułów ... 683
Rozwijanie i zwijanie ... 685
Wyświetlanie artykułów ... 688
Korzystanie z klasy wezel_drzewa ... 689
Przeglądanie pojedynczych artykułów ... 695
Dodawanie nowych artykułów ... 697
Rozszerzenia ... 703
Wykorzystanie istniejącego systemu ... 704
W następnym rozdziale ... 704

Rozdział 30. Tworzenie dokumentów spersonalizowanych w formacie PDF ... 705
Problem ... 705
Ocena formatów dokumentów ... 706
Papier ... 706
ASCII ... 707
HTML ... 707
Formaty edytorów tekstu ... 707
Format RTF ... 708
PostScript ... 709
Format PDF ... 710
Składniki rozwiązania ... 710
System pytań i odpowiedzi ... 711
Oprogramowanie generujące dokumenty ... 711
Przegląd rozwiązania ... 713
Zadawanie pytań ... 714
Ocena odpowiedzi ... 716
Tworzenie certyfikatu RTF ... 718
Tworzenie certyfikatu PDF z szablonu ... 722
Generowanie dokumentu PDF za pomocą PDFlib ... 725
Skrypt „Witaj świecie” dla PDFlib ... 725
Tworzenie certyfikatu za pomocą PDFlib ... 729
Problemy związane z nagłówkami ... 736
Rozwijanie projektu ... 736
Propozycje dalszych lektur ... 736

Rozdział 31. Korzystanie z usług sieciowych za pomocą XML i SOAP 737
Problem 737
Podstawy XML 738
Podstawy usług sieciowych 742
SOAP 742
WSDL 743
Składniki rozwiązania 743
Konstrukcja koszyka na zakupy 744
Korzystanie z interfejsu usług sieciowych Amazon.com 744
Wczytywanie dokumentów XML 745
Korzystanie z SOAP za pomocą PHP 745
Buforowanie 745
Opis rozwiązania 746
Aplikacja główna 750
Wyświetlanie listy książek z danej kategorii 754
Tworzenie obiektu klasy WynikiWyszukiwania 757
Przesyłanie dokumentów XML 765
Korzystanie z protokołu SOAP 770
Buforowanie danych 772
Konstrukcja koszyka na zakupy 774
Przejście do kasy na witrynie Amazon.com 777
Instalacja kodu źródłowego 778
Kierunki rozwoju 778
Literatura 778

Dodatki 779

Dodatek A Instalacja PHP4 i MySQL 781
Uruchamianie PHP jako CGI lub moduł serwera 782
Instalacja Apache, PHP i MySQL w systemie UNIX 782
Instalacja przy użyciu binariów 782
Instalacja przy użyciu kodów źródłowych 783
Plik httpd.conf — informacje końcowe 790
Czy obsługa PHP działa poprawnie? 790
Czy SSL działa poprawnie? 792
Instalacja Apache, PHP i MySQL w systemie Windows 793
Instalacja MySQL w systemie Windows 793
Instalacja serwera Apache w systemie Windows 797
Instalacja PHP w systemie Windows 799
Instalowanie PEAR 803
Inne konfiguracje 804

Dodatek B Zasoby internetowe 805
Zasoby poświęcone PHP 805
Zasoby poświęcone MySQL i SQL 807
Zasoby poświęcone serwerowi Apache 807
Zasoby poświęcone tworzeniu stron WWW 808

Skorowidz 809

Rozdział 8.

Tworzenie internetowej bazy danych

W rozdziale tym przedstawimy sposób tworzenia bazy danych MySQL, przeznaczonej do udostępniania na stronach WWW.

W tym rozdziale zostaną poruszone następujące zagadnienia:

- tworzenie bazy danych,
- użytkownicy i przywileje,
- wprowadzenie do systemu przywilejów,
- tworzenie tabel bazy danych,
- typy kolumn w MySQL.

Również i w tym rozdziale posłużymy się znanym już przykładem księgarni internetowej „Książkorama". Przypomnijmy schemat jej bazy danych:

```
Klienci(KlientID, Nazwisko, Adres, Miejscowosc)
Zamowienia (ZamowienieID, KlientID, Wartosc, Data)
Ksiazki (ISBN, Autor, Tytul, Cena)
Pozycje_zamowione (ZamowienieID, ISBN, Ilosc)
Recenzje_ksiazek (ISBN, Recenzja)
```

Należy również przypomnieć, iż nazwy pól będących kluczami podstawowymi są podkreślone linią ciągłą, a nazwy pól będących kluczami obcymi — linią przerywaną.

W celu przeanalizowania materiału zawartego w tym rozdziale należy zapewnić sobie dostęp do serwera MySQL, co oznacza, że:

- Trzeba przeprowadzić instalację MySQL na serwerze WWW, obejmującą następujące czynności:
  - instalację plików,
  - rejestrację użytkownika na serwerze MySQL,
  - zdefiniowanie odpowiedniej ścieżki dostępu,
  - w razie konieczności uruchomienie aplikacji `mysql_install_db`,

- określenie hasła administratora,
- usunięcie użytkownika anonimowego,
- uruchomienie serwera MySQL i skonfigurowanie go w taki sposób, by uruchamiał się automatycznie.

Po wykonaniu wymienionych czynności można przystąpić do lektury tego rozdziału. Jeśli jednak instalacja nie powiodła się, należy kierować się instrukcjami zawartymi w dodatku A.

Ewentualne błędy, które mogą wystąpić w trakcie wykonywania czynności opisywanych w tym rozdziale, spowodowane będą prawdopodobnie nieprawidłową pracą serwera MySQL. W takim przypadku należy ponownie przeanalizować czynności opisane powyżej oraz przedstawione w dodatku A w celu upewnienia się, iż MySQL został prawidłowo zainstalowany.

- Należy zapewnić sobie dostęp do serwera MySQL zainstalowanego na komputerze znajdującym się w miejscu pracy czytelnika, udostępnianego przez dostawcę usług internetowych na zdalnym komputerze itp.

  W takim przypadku wykonanie zawartych w tym rozdziale przykładów lub utworzenie własnej bazy danych wymaga zarejestrowania przez administratora odpowiedniego użytkownika oraz znajomości nadanego mu identyfikatora, hasła dostępu i nazwy przypisanej mu bazy danych.

  Nie jest konieczne zapoznanie się z treścią podrozdziałów dotyczących rejestrowania użytkowników i przydzielania im bazy danych, chyba że istnieje potrzeba bardziej precyzyjnego wyjaśnienia administratorowi przedstawionych wcześniej wymagań. Zwykli użytkownicy nie będą wszakże mogli samodzielnie tworzyć własnych użytkowników i baz danych.

Wszystkie przykłady zawarte w tym rozdziale były uruchamiane na serwerze MySQL w wersji 3.23.52. Niektóre wcześniejsze wersje serwera nie posiadają pewnych możliwości, dlatego też przed rozpoczęciem lektury należy zastąpić starszą wersję nową instalacją MySQL. Najnowszą wersję serwera MySQL można pobrać ze strony *http://mysql.com*.

Uwagi na temat użytkowania monitora MySQL

Nietrudno zauważyć, że w przykładach przedstawionych w dalszej części tego rozdziału oraz w rozdziale następnym każda komenda jest zakończona znakiem średnika (;). Informuje on serwer MySQL o konieczności wykonania zadanego polecenia. Opuszczenie średnika spowoduje brak reakcji serwera, o czym często zapominają użytkownicy, zwłaszcza niedoświadczeni.

Dzięki temu mechanizmowi możliwe jest również przejście do nowego wiersza przed zakończeniem wpisywania tekstu komendy, co pozwoliło na zwiększenie czytelności przykładów. Odnalezienie miejsca, w którym nastąpiło przejście do nowego wiersza, nie powinno sprawić żadnych problemów — MySQL wyświetli tam znak kontynuacji w formie strzałki przedstawionej poniżej:

```
mysql> grant select
    ->
```

Symbol ten oznacza, że MySQL czeka na wprowadzenie dalszej części polecenia. Za każdym razem, gdy zostanie naciśnięty klawisz Enter bez wcześniejszego wpisania znaku średnika, nastąpi przejście do nowego wiersza i wyświetlenie symbolu strzałki.

Warto również zapamiętać, że MySQL nie rozróżnia małych i wielkich liter w poleceniach języka SQL, może natomiast je rozróżniać w nazwach baz danych i tabel. Kwestia ta zostanie poruszona w dalszej części rozdziału.

Jak zalogować się do serwera MySQL

W celu zalogowania się do serwera MySQL należy przejść do wiersza poleceń systemu operacyjnego i wpisać następującą komendę:

```
> mysql -h nazwa_komputera -u identyfikator_uzytkownika -p
```

Znak zachęty może mieć inną formę, zależnie od używanego systemu operacyjnego i powłoki.

Polecenie `mysql` wywołuje monitor MySQL, mający postać wiersza poleceń łączącego klienta z serwerem.

Opcji `-h` używa się w celu wskazania komputera, do którego ma nastąpić połączenie (na maszynie tej pracuje serwer MySQL). Jeśli polecenie to ma być wykonane na tym samym komputerze, na którym znajduje się MySQL, to opcja `-h` oraz parametr *`nazwa_komputera`* mogą zostać pominięte. W przeciwnym wypadku konieczne jest wpisanie w miejsce parametru *`nazwa_komputera`* nazwy maszyny, na której uruchomiony jest serwer bazy danych.

Opcja `-u` jest stosowana w celu wskazania identyfikatora użytkownika, na którego następuje logowanie. Jeśli identyfikator ten nie zostanie podany, wówczas MySQL domyślnie użyje identyfikatora tego użytkownika, który jest aktualnie zalogowany do systemu operacyjnego.

Jeśli MySQL został zainstalowany na komputerze lub serwerze czytelnika, będzie on zmuszony zalogować się jako użytkownik `root` oraz samodzielnie utworzyć bazę danych, którą będzie wykorzystywał w trakcie uruchamiania zawartych w tym rozdziale przykładów. Przy pierwszym uruchomieniu serwera użytkownik `root` jest jedynym zarejestrowanym użytkownikiem.

W trakcie pracy na serwerze zainstalowanym na komputerze administrowanym przez inną osobę należy przy logowaniu użyć identyfikatora użytkownika, podanego przez administratora.

Opcja `-p` informuje serwer o logowaniu się z użyciem hasła. Może ona zostać pominięta, jeśli dla danego użytkownika hasło nie zostało ustanowione.

Jeżeli czytelnik loguje się jako użytkownik `root` bez konieczności podania hasła, zalecane jest określenie hasła dostępu zgodnie z instrukcjami zawartymi w dodatku A. Jego brak powoduje, iż system będzie niedostatecznie zabezpieczony.

Podawanie hasła w tym samym wierszu co polecenie `mysql` nie jest obowiązkowe. Jeżeli nie zostanie ono podane, to serwer MySQL sam o nie „zapyta". Pominięcie hasła jest właściwie lepszym rozwiązaniem, gdyż wpisane w wierszu poleceń będzie miało formę zwykłego tekstu, co umożliwi osobom postronnym jego wykrycie.

Po wpisaniu komendy podanej na początku podrozdziału wyświetlona zostanie odpowiedź w następującej formie:

```
Enter password: *****
```

(jeżeli to nie nastąpi, trzeba sprawdzić, czy serwer MySQL jest uruchomiony, a komenda `mysql` może być zrealizowana w bieżącej ścieżce dostępu).

Następnie należy podać hasło dostępu. Jeśli wszystko przebiegnie prawidłowo, powinien zostać wyświetlony komunikat podobny do poniższego:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 3.23.52-nt
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

Jeżeli czytelnik pracuje na własnym komputerze i komunikat nie będzie przypominał powyższego, wówczas należy upewnić się, czy uruchomiona jest aplikacja `mysql_ install_db` (jeżeli jest wymagana) i czy zostało określone i wpisane prawidłowo hasło dostępu dla użytkownika `root`.

Jeśli praca odbywa się na zdalnej maszynie, trzeba upewnić się, iż podane hasło jest prawidłowe.

Po zalogowaniu się do serwera kursor powinien znajdować się przy znaku zachęty, umożliwiając stworzenie nowej bazy danych.

Czytelnik pracujący na własnym komputerze powinien stosować się do instrukcji podanych w dalszej części rozdziału.

Czytelnik korzystający z maszyny administrowanej przez inną osobę powinien mieć utworzoną i przypisaną mu bazę danych, może więc od razu przejść do lektury podrozdziału „Używanie odpowiedniej bazy danych". Można oczywiście zapoznać się z poprzedzającymi go podrozdziałami, nie będzie jednak możliwości (a przynajmniej nie powinno być) uruchamiania wyszczególnionych w nich poleceń.

Tworzenie baz i rejestrowanie użytkowników

System baz danych MySQL jest w stanie obsługiwać wiele baz danych jednocześnie. Zazwyczaj jedna aplikacja współpracuje z jedną bazą danych. Dla księgarni „Książkorama" baza ta nosi nazwę `Ksiazki`.

Tworzenie bazy danych

To czynność najprostsza. W wierszu poleceń MySQL należy wpisać:

```
mysql> create database nazwa_bazy;
```

W miejsce *nazwa_bazy* należy podać nazwę bazy danych, która ma zostać utworzona. W naszym przykładzie jest to baza o nazwie `Ksiazki`.

I to wszystko! Otrzymana odpowiedź powinna wyglądać mniej więcej tak:

```
Query OK, 1 row affected (0.06 sec)
```

Oznacza to, że operacja została wykonana bez błędów. Jeśli taki komunikat się nie pojawi, należy upewnić się, że na końcu komendy został wpisany znak średnika. Informuje on serwer MySQL o zakończeniu wpisywania polecenia i o konieczności jego wykonania.

Użytkownicy i przywileje

System MySQL może obsługiwać wielu użytkowników. Użytkownik `root` powinien być wykorzystywany w zasadzie tylko do celów administracyjnych, co jest podyktowane względami bezpieczeństwa. Dla każdego użytkownika, który będzie pracował w systemie, trzeba utworzyć konto i nadać mu hasło dostępu. Nie muszą one być takie same, jak identyfikator użytkownika i hasło wykorzystywane do zalogowania się do systemu operacyjnego. Ta sama zasada odnosi się do użytkownika `root`. Pożądane jest stosowanie różnych haseł dostępu w razie logowania się do systemu operacyjnego i serwera MySQL, szczególnie w przypadku użytkownika `root`.

Nadawanie hasła nowym użytkownikom nie jest obowiązkowe, jednak ze względów bezpieczeństwa pożądane jest, aby każdy użytkownik posiadał własne hasło dostępu.

W przypadku tworzenia internetowej bazy danych wskazane jest utworzenie co najmniej jednego użytkownika, który będzie wykorzystywany tylko przez daną aplikację bazodanową.

Można by zadać pytanie o cel tej operacji. Odpowiedź związana jest z systemem przywilejów.

Wprowadzenie do systemu przywilejów MySQL

Jedną z najważniejszych cech MySQL jest obsługa wyrafinowanego systemu przywilejów.

Przywilej to inaczej posiadane przez określonego użytkownika prawo do wykonania określonego polecenia na określonym obiekcie. Idea przywilejów jest zbliżona do koncepcji praw dostępu do plików.

Rejestrując nowego użytkownika MySQL, należy nadać mu odpowiednie przywileje w celu wyszczególnienia czynności, które będzie on mógł wykonać w systemie.

Zasada najmniejszego przywileju

Zasada najmniejszego przywileju jest stosowana w celu zwiększenia bezpieczeństwa systemu komputerowego. Jest to podstawowa, ale bardzo ważna zasada, o której niestety często się zapomina. Brzmi ona następująco:

> Użytkownik (lub proces) powinien posiadać minimalny zbiór przywilejów potrzebnych do wykonania przypisanego mu zadania.

Zasadę tę należy stosować wszędzie, nie tylko w odniesieniu do MySQL. Na przykład do wysłania zapytania do bazy danych ze strony WWW użytkownik nie będzie potrzebował wszystkich przywilejów, które posiada użytkownik `root`. Należy więc utworzyć nowego użytkownika i nadać mu tylko przywileje niezbędne do uzyskania dostępu do bazy danych.

Rejestrowanie użytkowników: polecenie GRANT

Polecenia `GRANT` i `REVOKE` służą do nadawania i odbierania użytkownikom MySQL praw na czterech poziomach uprzywilejowania. Wyróżniamy następujące poziomy przywilejów:

- globalny,
- baza danych,
- tabela,
- kolumna.

W dalszej części rozdziału wyjaśnimy, który z nich i kiedy należy stosować.

Polecenie GRANT służy do tworzenia nowych użytkowników i nadawania im przywilejów. Składnia tego polecenia jest następująca:

```
GRANT przywileje [kolumny]
ON obiekt
TO identyfikator_uzytkownika [IDENTIFIED BY 'haslo']
[WITH GRANT OPTION]
```

Klauzule w nawiasach kwadratowych są opcjonalne. Poniżej zostaną omówione poszczególne parametry tego polecenia.

Pierwszy z nich, *przywileje*, ma postać listy przywilejów oddzielonych przecinkami. MySQL posiada liczny zbiór przywilejów, opisanych w następnym podrozdziale.

Parametr *kolumny* jest opcjonalny. Używa się go w celu wskazania kolumn, do których podane przywileje zostaną zastosowane. Można podać nazwę pojedynczej kolumny lub listę nazw kolumn oddzielonych przecinkami.

Parametr *obiekt* wskazuje bazę lub tabelę, do której zostaną zastosowane podane przywileje. W celu nadania przywilejów na wszystkie bazy danych w systemie *obiekt* powinien przyjąć wartość *.*. Wówczas przywileje zostały nadane na poziomie *globalnym*. Ten sam efekt można osiągnąć, przypisując parametrowi *obiekt* wartość *, jeśli nie jest używana żadna baza danych.

Częściej jednak wskazuje się wszystkie tabele w bazie, wpisując *nazwa_bazy*.*, konkretną tabelę (*nazwa_bazy.nazwa_tabeli*) lub pojedyncze kolumny w danej tabeli poprzez wpisanie *nazwa_bazy.nazwa_tabeli* i nadanie odpowiedniej wartości parametrowi *kolumny*. Za pomocą tych metod nadaje się przywileje na pozostałych trzech poziomach uprzywilejowania, odpowiednio: *baza danych*, *tabela* i *kolumna*. Jeżeli w trakcie wykonywania tego polecenia używana jest konkretna baza danych, to podanie samej nazwy tabeli będzie zinterpretowane jako nadanie przywilejów tabeli o tej nazwie, znajdującej się w używanej bazie danych.

Parametr *identyfikator_uzytkownika* powinien wskazywać identyfikator, za pomocą którego użytkownik loguje się do serwera MySQL. Oczywiście nie musi być to ten sam, który jest używany przy logowaniu się do systemu operacyjnego. Wartość tego parametru może również zawierać nazwę komputera. Możliwość tę wykorzystuje się na przykład do odróżnienia użytkownika `laura` (traktowanego jako `laura@localhost`) od użytkownika `laura@gdzieindziej.com`. Jest to szczególnie użyteczne w przypadku, gdy użytkownicy z różnych domen mają te same identyfikatory. Poza tym metoda ta zwiększa poziom bezpieczeństwa systemu, gdyż pozwala określić, z jakiej domeny konkretny użytkownik może się zalogować, a nawet do jakich tabel lub baz ma on dostęp z określonej lokalizacji.

Wartością parametru *haslo* jest hasło dostępu, którym powinien posługiwać się wskazany użytkownik w trakcie logowania do serwera. Stosuje się tutaj standardowe zasady doboru haseł. Kwestie bezpieczeństwa zostaną omówione w dalszej części książki, jednak należy zaznaczyć, że hasło nie powinno być łatwe do odgadnięcia. Niewskazane jest używanie słów, które można znaleźć w słowniku ani nie powinno być takie samo, jak identyfikator użytkownika. Najlepszym rozwiązaniem jest nadanie hasła składającego się z liter małych i wielkich oraz znaków nie będących literami.

Dodanie opcji `WITH GRANT OPTION` spowoduje, że wskazany użytkownik będzie mógł nadawać innym użytkownikom takie przywileje, jakie sam posiada.

Informacje dotyczące nadanych przywilejów zapamiętywane są w czterech tabelach systemowych, znajdujących się w bazie danych o nazwie `mysql`. Tabele te noszą nazwy: `mysql.user`, `mysql.db`, `mysql.tables_priv` oraz `mysql.columns_priv`; każda z nich odpowiada jednemu z czterech poziomów uprzywilejowania opisanych wcześniej. Zamiast wykorzystywać polecenie `GRANT` można zmieniać dane bezpośrednio w wymienionych tabelach. Zagadnienie to zostanie opisane w rozdziale 11.

Typy i poziomy przywilejów

MySQL wykorzystuje trzy typy przywilejów:

- przywileje nadawane zwykłym użytkownikom,
- przywileje dla administratorów,
- przywileje specjalne.

Każdemu użytkownikowi można przyporządkować przywileje dowolnego typu, najrozsądniejsze jednak jest nadawanie przywilejów dla administratorów tylko administratorom systemu, czyli zgodnie ze wspomnianą wcześniej zasadą najmniejszego przywileju.

Użytkownikom należy nadawać takie przywileje, które umożliwią korzystanie tylko z potrzebnych im baz i tabel. Nikomu, z wyjątkiem administratora, nie należy udostępniać systemowej bazy `mysql`, gdyż w niej właśnie przechowywane są identyfikatory wszystkich użytkowników, ich hasła dostępu itp. (szerzej zagadnienie to jest opisane w rozdziale 11.).

Przywileje przeznaczone dla zwykłych użytkowników odwołują się bezpośrednio do konkretnych rodzajów poleceń SQL i określają, czy dany użytkownik może wykonywać polecenia tego typu. Polecenia języka SQL zostaną szerzej omówione w następnym rozdziale, teraz ograniczymy się tylko do krótkiego opisu (tabela 8.1). Kolumna „Zastosowanie" wskazuje obiekty, którym może być nadany określony typ przywilejów.

Tabela 8.1. *Przywileje dla użytkowników*

Przywilej	Zastosowanie	Opis
SELECT	tabele, kolumny	Pozwala na wyszukiwanie wierszy (rekordów) z tabel.
INSERT	tabele, kolumny	Pozwala na wstawianie nowych wierszy do tabel.
UPDATE	tabele, kolumny	Pozwala na zmianę wartości wierszy zapisanych w tabeli.
DELETE	tabele	Pozwala na usuwanie z tabeli istniejących wierszy.
INDEX	tabele	Pozwala na tworzenie i usuwanie indeksów w poszczególnych tabelach.
ALTER	tabele	Pozwala na dokonywanie zmian w strukturze istniejących tabel, np. dodawanie nowych kolumn, zmianę nazw kolumn lub tabel, zmianę typu danych istniejących kolumn.
CREATE	bazy danych, tabele	Pozwala na tworzenie nowych tabel i baz danych. Jeżeli w ramach polecenia GRANT podano nazwę konkretnej tabeli lub kolumny, to użytkownik ma prawo utworzyć tabelę lub kolumnę tylko o tej nazwie, a więc najpierw będzie zmuszony ją usunąć.
DROP	bazy danych, tabele	Pozwala na usuwanie baz lub tabel.

Większość przywilejów przeznaczonych dla zwykłych użytkowników nie powoduje zmniejszenia bezpieczeństwa systemu. Przywilej ALTER może być wykorzystywany do obchodzenia systemu przywilejów poprzez zmianę nazw tabel, jednak jest on potrzebny większości użytkowników. Ochrona systemu jest zawsze rezultatem kompromisu między jego użytecznością a poziomem zabezpieczeń. Administrator powinien samodzielnie podejmować decyzję o nadaniu, lub nie, przywileju ALTER zwykłym użytkownikom. Zazwyczaj jednak przywilej ten jest nadawany.

Oprócz przywilejów opisanych w tabeli 8.1 istnieje również przywilej REFERENCES (obecnie nie jest on praktycznie używany) oraz przywilej GRANT, nadawany częściej poprzez dodanie opcji WITH GRANT OPTION niż przez dołączanie go do listy wartości parametru *przywileje*.

W tabeli 8.2 opisano przywileje przeznaczone dla administratorów.

Tabela 8.2. *Przywileje dla administratorów*

Przywilej	Opis
RELOAD	Pozwala na powtórne załadowanie tabel zawierających informacje na temat praw dostępu oraz na odświeżenie przywilejów, listy nazw łączących się komputerów, dziennika zdarzeń i tabel.
SHUTDOWN	Umożliwia zakończenie pracy serwera MySQL.
PROCESS	Pozwala na śledzenie procesów wykonywanych przez serwer i ich przerywanie.
FILE	Pozwala na wczytywanie danych z plików do tabel i odwrotnie.

Istnieje możliwość nadania opisanych przywilejów użytkownikom nie mającym statusu administratora, jednak należy tego dokonywać z zachowaniem najwyższej ostrożności. Zwykłemu użytkownikowi nie powinno się zwłaszcza nadawać przywilejów RELOAD, SHUTDOWN i PROCESS.

Z przywilejem FILE sprawa wygląda nieco inaczej. Jest on bardzo przydatny dla użytkowników, gdyż możliwość wstawiania danych bezpośrednio z plików do tabel pozwala zaoszczędzić czas, konieczny na ręczne wpisywanie kolejnych wartości. Z drugiej strony jednak może zostać wykorzystany do załadowania wszelkich plików, które widzi serwer MySQL, np. pliku bazy należącej do innego użytkownika czy pliku zawierającego hasła dostępu. Przywilej ten powinien więc być nadawany z rozwagą lub też administrator powinien sam dokonywać ładowania danych z pliku do wskazanej przez użytkownika tabeli.

MySQL posiada również dwa specjalne przywileje, przedstawione w tabeli 8.3.

Tabela 8.3. *Przywileje specjalne*

Przywilej	Opis
ALL	Nadaje wszystkie przywileje opisane w tabelach 8.1 i 8.2. Jest on równoważny przywilejowi ALL PRIVILEGES.
USAGE	Nie nadaje żadnych przywilejów. Powoduje zarejestrowanie użytkownika i pozwala mu na zalogowanie się, lecz jakiekolwiek inne czynności są dla niego niedostępne. Odpowiednie przywileje nadawane są zwykle później.

Polecenie REVOKE

Przeciwieństwem GRANT jest polecenie REVOKE, używane w celu odebrania użytkownikowi określonych przywilejów. Jego składnia jest bardzo podobna do składni polecenia GRANT:

```
REVOKE przywileje [kolumny]
ON obiekt
FROM identyfikator_uzytkownika
```

Jeżeli do polecenia GRANT dołączono opcję WITH GRANT OPTION, przywilej ten można cofnąć w następujący sposób:

```
REVOKE GRANT OPTION
ON obiekt
FROM identyfikator_uzytkownika
```

Przykłady użycia poleceń GRANT i REVOKE

Aby zarejestrować użytkownika mającego status administratora, należy wpisać:

```
mysql> grant all
    -> on *
    -> to fred identified by 'mnb123'
-> with grant option;
```

Polecenie to spowoduje nadanie wszystkich przywilejów na wszystkie bazy danych użytkownikowi o identyfikatorze Fred, posługującemu się hasłem „mnb123", oraz umożliwi nadanie dowolnego przywileju innym użytkownikom.

Niekiedy konieczne okazuje się wyeliminowanie tego użytkownika. Można to zrobić w następujący sposób:

```
mysql> revoke all
    -> on *
    -> from fred;
```

Zarejestrujmy teraz użytkownika, który nie będzie posiadał żadnych przywilejów:

```
mysql> grant usage
    -> on ksiazki.*
    -> to zosia identified by 'magic123';
```

Po rozmowie z Zosią wiadomo już, w jaki sposób chce korzystać z bazy, można więc nadać jej odpowiednie przywileje:

```
mysql> grant select, insert, update, delete, index, alter, create, drop
    -> on ksiazki.*
    -> to zosia;
```

Jak widać, nie ma już potrzeby podawania hasła dostępu, jakim posługuje się Zosia.

Jeżeli administrator dojdzie do wniosku, że Zosia wykonała już część swoich zadań, może ograniczyć nadane jej przywileje:

```
mysql> revoke alter, create, drop
    -> on ksiazki.*
    -> from zosia;
```

Kiedy Zosia nie będzie już potrzebowała dostępu do bazy danych, można odebrać jej wszystkie pozostałe przywileje:

```
mysql> revoke all
    -> on ksiazki.*
    -> from zosia;
```

Rejestrowanie użytkownika łączącego się z Internetu

Konieczne jest zarejestrowanie użytkownika, który łączy się z bazą danych poprzez stronę WWW zawierającą skrypty PHP. Również w tym przypadku należy zastosować zasadę najmniejszego przywileju, trzeba więc rozważyć, jakie operacje ma wykonywać skrypt PHP.

W większości przypadków wystarczą przywileje `SELECT`, `INSERT`, `DELETE` i `UPDATE`. Nadaje się je w następujący sposób:

```
mysql> grant select, insert, delete, update
    -> on ksiazki.*
    -> to ksiazkorama identified by 'ksiazkorama';
```

Oczywiście hasło powinno być bardziej skomplikowane niż podane w przykładzie.

Użytkownik korzystający z usług zewnętrznej firmy internetowej otrzyma zapewne szersze przywileje na pracę z przydzieloną mu bazą danych. Identyfikator użytkownika i jego hasło dostępu będą prawdopodobnie umożliwiały zarówno wydawanie poleceń serwerowi MySQL (tworzenie tabel itp.), jak i łączenie się z bazą z poziomu strony WWW (np. wyszukiwanie danych w tabeli). Rozwiązanie takie mogłoby obniżyć nieznacznie stopień zabezpieczenia systemu, zalecane jest więc zarejestrowanie nowego użytkownika z następującym zbiorem przywilejów:

```
mysql> grant select, insert, update, delete, index, alter, create, drop
    -> on ksiazki.*
    -> to ksiazkorama identified by 'ksiazkorama';
```

Utwórz więc drugą wersję użytkownika, gdyż będziemy jej potrzebować w następnych sekcjach.

Wylogowanie się użytkownika root

Wylogowanie się z serwera MySQL jest dokonywane za pomocą polecenia `quit`. Zalecane jest ponowne zalogowanie się jako użytkownik internetowy (zarejestrowany w poprzednim punkcie) i sprawdzenie poprawności działania bazy danych.

Używanie odpowiedniej bazy danych

Przed przystąpieniem do lektury tego podrozdziału czytelnik powinien być już zalogowany do serwera MySQL na koncie zwykłego użytkownika, założonym w poprzednim podrozdziale lub też utworzonym przez administratora systemu.

Pierwszą czynnością, jaką należy wykonać, jest wskazanie bazy danych, która zostanie wykorzystana. Służy do tego następujące polecenie:

```
mysql> use nazwa_bazy;
```

gdzie *nazwa_bazy* jest nazwą bazy danych.

Tę samą operację można wykonać, podając nazwę bazy danych w trakcie logowania się do serwera:

```
> mysql nazwa_bazy -h nazwa_komputera -u identyfikator_uzytkownika -p
```

W omawianym przykładzie wykorzystywana będzie baza danych `Ksiazki`:

```
mysql> use ksiazki;
```

Po wykonaniu tego polecenia MySQL powinien zwrócić następujący komunikat:

```
Database changed
```

Jeżeli przed przystąpieniem do pracy nie zostanie wybrana żadna baza danych, serwer wyświetli komunikat o błędzie:

```
ERROR 1046: No Database Selected
```

Tworzenie tabel bazy danych

Następnym etapem tworzenia bazy danych jest konstrukcja tabel. Służy do tego polecenie języka SQL `CREATE TABLE`, którego składnia przedstawia się następująco:

```
CREATE TABLE nazwa_tabeli(kolumny)
```

Parametr *nazwa_tabeli* powinien określać nazwę tabeli, która ma zostać utworzona. Z kolei parametr *kolumny* powinien mieć postać listy kolumn oddzielonych przecinkami, jakie będzie zawierać nowa tabela.

Każda z kolumn musi być określona przez podanie jej nazwy i typu danych.

Przypomnijmy schemat bazy danych księgarni „Książkorama":

Klienci(KlientID, Nazwisko, Adres, Miejscowosc)

Zamowienia (ZamowienieID, KlientID, Wartosc, Data)

Ksiazki (ISBN, Autor, Tytul, Cena)

Pozycje_zamowione (ZamowienieID, ISBN, Ilosc)

Recenzje_ksiazek (ISBN, Recenzja)

Na listingu 8.1 przedstawione są polecenia SQL, za pomocą których zostaną utworzone powyższe tabele (przy założeniu, że istnieje już baza danych `Ksiazki`). Kod ten znajduje się również w folderze *rozdzial_08* w pliku o nazwie *ksiazkorama.sql* (przykłady są dostępne na płycie CD dołączonej do książki).

Listing 8.1. *ksiazkorama.sql — kod SQL tworzy tabele w bazie danych księgarni „Książkorama"*

```
create table klienci
( klientid int unsigned not null auto_increment primary key,
  nazwisko char(30) not null,
  adres char(40) not null,
  miejscowosc char(20) not null
);

create table zamowienia
( zamowienieid int unsigned not null auto_increment primary key,
  klientid int unsigned not null,
  wartosc float(6,2),
  data date not null
);

create table ksiazki
( isbn char(13) not null primary key,
  autor char(30),
  tytul char(60),
  cena float(4,2)
);

create table pozycje_zamowione
( zamowienieid int unsigned not null,
  isbn char(13) not null,
```

```
  ilosc tinyint unsigned,

  primary key (zamowienieid, isbn)

);

create table recenzje_ksiazek
( isbn char(13) not null primary key,
  recenzja text
);
```

Kod SQL zawarty w pliku *ksiazkorama.sql* zostanie wykonany po wpisaniu następującego polecenia (zakładamy przy tym, że plik *ksiazkorama.sql* ma tę samą lokalizację, co aplikacja `mysql`):

```
> mysql -h nazwa_komputera -u ksiazkorama -D ksiazki -p < ksiazkorama.sql
```

(Pamiętaj, by parametr nazwa_komputera zastąpić nazwą swego komputera).

Mechanizm przekierowania do pliku zawierającego kod SQL jest bardzo poręczny, gdyż pozwala na edycję kodu za pomocą dowolnego edytora tekstowego przed jego wykonaniem.

Pojedyncza tabela jest tworzona za pomocą odrębnego polecenia `CREATE TABLE`. Jak widać, każda z tabel zawiera kolumny wyszczególnione w schemacie bazy danych, utworzonym w poprzednim rozdziale. Ponadto każdej kolumnie nadano nazwę oraz przypisano typ danych, jakie będą w niej przechowywane. Niektóre kolumny są opisywane przez dodatkowe atrybuty, objaśnione w następnym punkcie.

Znaczenie dodatkowych atrybutów kolumn

`NOT NULL` oznacza, że pole, przy którym ten atrybut stoi, musi w każdym wierszu tabeli mieć nadaną jakąś wartość. Jeżeli przy opisie nowej kolumny atrybut został pominięty, wówczas jej pola mogą być puste (`NULL`).

`AUTO_INCREMENT` jest specjalnym atrybutem nadawanym kolumnom przechowującym wartości całkowitoliczbowe. Jeżeli do tabeli zostanie wpisany nowy rekord, w którym wartość pola z atrybutem `AUTO_INCREMENT` będzie nieokreślona, wówczas serwer MySQL automatycznie wstawi w to miejsce unikalny identyfikator — będzie nim liczba całkowita o wartości równej dotychczasowej maksymalnej wartości w tej kolumnie i powiększonej o jeden. W każdej tabeli może występować najwyżej jedna kolumna z tym atrybutem. Kolumny z atrybutem `AUTO_INCREMENT` muszą być indeksowane.

Atrybut `PRIMARY KEY` oznacza, że wskazana kolumna jest kluczem podstawowym tabeli, a wartości w niej zapisywane muszą być unikalne. MySQL automatycznie indeksuje takie kolumny. Zauważmy, że w pliku *ksiazkorama.sql* wszystkim kolumnom z atrybutem `AUTO_INCREMENT` (np. `KlientID` w tabeli `Klienci`) przypisano również atrybut `PRIMARY KEY`, dzięki czemu został spełniony wymóg indeksowania pól z atrybutem `AUTO_INCREMENT`.

Atrybut `PRIMARY KEY` może być wyszczególniony zaraz za nazwą nowej kolumny wyłącznie w przypadku, gdy tylko ta jedna kolumna jest kluczem podstawowym tabeli. Inny sposób określenia klucza podstawowego został zaprezentowany dla tabeli `Pozycje_`

zamowione. Wykorzystanie omawianej metody jest wymuszone tym, że w skład klucza podstawowego tabeli Pozycje_zamowione wchodzi nie jedna, ale dwie kolumny jednocześnie.

Atrybut UNSIGNED wpisany za identyfikatorem typu całkowitoliczbowego oznacza, że kolumna może zawierać tylko wartości nieujemne.

Typy kolumn

Rozważmy przykład pierwszej utworzonej tabeli:

```
create table klienci
( klientid int unsigned not null auto_increment primary key,
  nazwisko char(30) not null,
  adres char(40) not null,
  miejscowosc char(20) not null
);
```

Tworząc nową tabelę, należy każdej jej kolumnie przypisywać odpowiedni typ danych.

Według schematu tabela klienci składa się z czterech kolumn. Pierwsza z nich, KlientID, jest kluczem podstawowym o wartościach typu całkowitoliczbowego (typ int). Co więcej, wszystkie pola ID (KlientID, ZamowienieID) mogą mieć tylko wartości nieujemne. Wykorzystano również cechy atrybutu AUTO_INCREMENT, dzięki któremu MySQL sam dba o spełnienie powyższych wymogów — mamy więc jeden kłopot z głowy.

Pozostałe kolumny będą przechowywać dane mające postać łańcuchów znaków — nadano im więc typ char o określonej dla każdej kolumny długości łańcucha, zawartej w nawiasach okrągłych. Na przykład w polu Nazwisko można zapisywać łańcuchy zawierające nie więcej niż 30 znaków.

Nawet jeśli nazwisko klienta liczy mniej niż 30 liter wartość pola Nazwisko, zawsze będzie zajmować tyle pamięci, ile trzeba do zapisania 30 znaków. MySQL automatycznie doda do nazwiska tyle znaków spacji, ile brakuje do wykorzystania całej przydzielonej polu pamięci. Alternatywą dla typu char jest varchar — wartości pól tego typu zajmują zawsze o jeden bajt więcej niż trzeba do zapamiętania podanego łańcucha. Stosowanie pól typu varchar oszczędza więc pamięć, może jednak negatywnie wpływać na wydajność całego systemu.

Dla prawdziwych klientów, ich nazwisk i adresów rozmiar odpowiednich pól może okazać się niewystarczający.

Większość kolumn została zadeklarowana jako NOT NULL, dzięki czemu wydajność pracy systemu może nieznacznie się polepszyć. Zagadnienia dotyczące optymalizacji pracy serwera MySQL zostaną poruszone w rozdziale 11.

W obrębie niektórych poleceń CREATE TABLE zawarte są elementy, które nie zostały jeszcze omówione. Rozważmy kolejny przykład:

```
create table zamowienia
( zamowienieid int unsigned not null auto_increment primary key,
  klientid int unsigned not null,
  wartosc float(6,2),
```

```
  data date not null
);
```

Kolumna `Wartosc` jest zadeklarowana jako liczba zmiennoprzecinkowa typu `float`. Dla większości typów liczbowych zmiennoprzecinkowych istnieje możliwość określenia maksymalnej liczby cyfr, jakie będą wyświetlane, oraz ilości miejsc znaczących po przecinku. W przykładzie księgarni internetowej wartość zamówień jest określana w złotówkach, zadeklarowano więc, iż ma ona nie więcej niż sześć cyfr plus dwie cyfry znaczące po przecinku.

Dane przechowywane w kolumnie `Data` mają typ `date`.

W rozważanej tabeli wszystkie kolumny, oprócz kolumny `Wartosc`, są zadeklarowane jako `NOT NULL`. Dlaczego? Otóż wpisując nowe zamówienie należy najpierw zapisać je w tabeli `Zamowienia`, następnie dodać odpowiednie rekordy do tabeli `Pozycje_zamowione`, po czym obliczyć wartość całego zamówienia. W momencie wpisywania nowego zamówienia jego wartość nie zawsze jest znana, dlatego wartości pola `Wartosc` mogą być nieokreślone.

W podobny sposób jak `Zamowienia` jest tworzona tabela `Ksiazki`:

```
create table ksiazki
( isbn char(13) not null primary key,
  autor char(30),
  tytul char(60),
  cena float(4,2)
);
```

W tym przypadku nie ma potrzeby tworzenia klucza podstawowego, ponieważ każda publikacja ma własny, unikalny numer ISBN. Nie licząc kolumny `ISBN`, wszystkie pozostałe kolumny tej tabeli mogą posiadać wartości `NULL`. Można bowiem wyobrazić sobie sytuację, w której znany jest tylko numer ISBN książki, zanim jeszcze zostanie ujawniony jej tytuł, autor i cena.

Na przykładzie tabeli `Pozycje_zamowione` zademonstrowano sposób deklarowania kluczy podstawowych składających się z więcej niż jednej kolumny:

```
create table pozycje_zamowione
( zamowienieid int unsigned not null,
  isbn char(13) not null,
  ilosc tinyint unsigned,

  primary key (zamowienieid, isbn)

);
```

Pole `Ilosc`, w którym zapamiętywana jest liczba zamówionych egzemplarzy jednej książki, jest typu `TINYINT UNSIGNED`, a więc może przyjmować wartości liczb całkowitych z zakresu od 0 do 255.

Jak już wspomnieliśmy, klucze podstawowe składające się z więcej niż jednej kolumny należy deklarować za pomocą klauzuli `PRIMARY KEY`. Przykład jej użycia zaprezentowano powyżej.

Ostatnie polecenie z pliku *ksiazkorama.sql* powoduje utworzenie tabeli `Recenzje_ksiazek`:

```
create table recenzje_ksiazek
```

```
( isbn char(13) not null primary key,
  recenzja text
);
```

Użyto w nim nowego typu danych, który nie został jeszcze omówiony — to typ `text`. Jest on stosowany w przypadku dłuższych tekstów, np. artykułów prasowych. Istnieje kilka jego wariantów, które zostaną szczegółowo opisane w dalszej części tego rozdziału.

Dla jeszcze lepszego zrozumienia procesu tworzenia tabel podane zostaną zasady nadawania nazw kolumnom oraz ogólne informacje na temat identyfikatorów. Najpierw jednak powrócimy na chwilę do utworzonej bazy danych.

Rzut oka na bazę danych — polecenia SHOW i DESCRIBE

Po zalogowaniu się do serwera MySQL i wybraniu bazy `Ksiazki` można sprawdzić, jakie tabele wchodzą w jej skład, wpisując następujące polecenie:

```
mysql> show tables;
```

Serwer wyświetli wówczas listę wszystkich tabel wybranej bazy danych:

```
+------------------+
| Tables_in_ksiazki |
+------------------+
| klienci          |
| ksiazki          |
| pozycje_zamowione |
| recenzje_ksiazek  |
| zamowienia       |
+------------------+
5 rows in set (0.06 sec)
```

Polecenia `SHOW` można również użyć do wyświetlenia wszystkich baz danych udostępnianych przez serwer:

```
mysql> show databases;
```

Szczegółowe informacje na temat konkretnej tabeli są dostępne po użyciu polecenia `DESCRIBE`:

```
mysql> describe ksiazki;
```

MySQL wyświetli wszystkie informacje, które należało podać przy tworzeniu tabeli:

```
+-------+------------+------+-----+---------+-------+
| Field | Type       | Null | Key | Default | Extra |
+-------+------------+------+-----+---------+-------+
| isbn  | char(13)   |      | PRI |         |       |
| autor | char(30)   | YES  |     | NULL    |       |
| tytul | char(60)   | YES  |     | NULL    |       |
| cena  | float(4,2) | YES  |     | NULL    |       |
+-------+------------+------+-----+---------+-------+
4 rows in set (0.05 sec)
```

Oba polecenia są szczególnie przydatne do przypomnienia sobie np. typu danych przechowywanych w konkretnej kolumnie czy przeanalizowania struktury bazy utworzonej przez innego projektanta.

Identyfikatory MySQL

MySQL rozróżnia cztery typy identyfikatorów: bazy danych, tabele, kolumny (wszystkie są już czytelnikowi znane) oraz aliasy (zostaną one szerzej omówione w następnym rozdziale).

Bazy danych w MySQL odwzorowują nazwy katalogów w strukturze plików systemu operacyjnego, natomiast tabele odpowiadają pojedynczym plikom. Fakt ten ma bezpośredni wpływ na zasady nazewnictwa oraz warunkuje rozróżnianie (lub nie) wielkich i małych liter w nazwach baz i tabel. Jeżeli system operacyjny rozróżnia wielkie i małe litery, wówczas czyni to również serwer baz danych (tak jest w przypadku systemu Unix), w przeciwnym razie wielkość liter nie będzie miała znaczenia (np. w przypadku Windows). W nazwach kolumn i aliasów wielkość liter nie odgrywa roli, jednak w obrębie jednego polecenia nazwy te muszą być zapisywane w taki sam sposób.

Warto wspomnieć, że lokalizacja katalogów i plików z danymi jest ustawiana w plikach konfiguracyjnych serwera MySQL. Można ją odnaleźć, wpisując w wierszu poleceń systemu operacyjnego następującą komendę:

```
mysqladmin variables
```

Szukamy zmiennej `datadir`.

Podsumowanie rozważań na temat identyfikatorów znajduje się w tabeli 8.4. Jedynym wyjątkiem, o którym należy wspomnieć, jest fakt, że identyfikatory MySQL nie mogą zawierać znaków o kodzie ASCII równym 0 i 255 (zresztą i tak nie wiadomo, w jakim celu miałyby zostać użyte).

Tabela 8.4. *Identyfikatory MySQL*

Typ identyfikatora	Długość maksymalna	Rozróżnianie wielkości liter	Znaki dozwolone
Baza danych	64	Tak jak system operacyjny	Wszystkie dozwolone przez system operacyjny w nazwie katalogów, z wyjątkiem znaku /
Tabela	64	Tak jak system operacyjny	Wszystkie dozwolone przez system operacyjne w nazwie plików, z wyjątkiem znaków / i .
Kolumna	64	Nie	Wszystkie
Alias	255	Nie	Wszystkie

Podane zasady ulegają jednak częstym zmianom.

Począwszy od wersji 3.23.6 serwera MySQL identyfikatory mogą zawierać słowa zarezerwowane i wszelkiego rodzaju znaki specjalne. Jedynym wymogiem jest wówczas konieczność zamknięcia ich w odwrotnych apostrofach (uzyskiwanych przez wciśnięcie klawisza tyldy w górnym lewym rogu większości klawiatur). Można na przykład napisać:

```
create database `create database`;
```

Wcześniejsze wersje systemu MySQL są pod tym względem bardziej restrykcyjne i nie dozwalają wykonać tak sformułowanego polecenia.

Oczywiście z tak szerokich możliwości należy korzystać z umiarem. To, że baza *może* nosić nazwę `create database`, wcale nie oznacza, że *powinna* zostać tak nazwana. Stosuje się tu standardową zasadę nazewnictwa, według której identyfikatory powinny mieć sensowne brzmienie.

Typy danych w kolumnach

W MySQL wyróżnia się trzy podstawowe typy danych, które mogą być przechowywane w kolumnach: liczbowy, daty i czasu oraz łańcuchowy. Każda z tych kategorii dzieli się na szereg podtypów. W tym podrozdziale scharakteryzujemy krótko wszystkie dostępne typy danych, natomiast ich wady i zalety omówimy w rozdziale 11.

Wielkość pamięci potrzebnej do przechowania jednej danej ściśle zależy od jej typu. Deklarując typ kolumny, należy kierować się następującą zasadą: zawsze wybiera się taki typ danych, który będzie zajmował najmniej pamięci i jednocześnie pozwoli na zapisanie wszystkich potrzebnych informacji.

W przypadku niektórych typów danych możliwe jest określenie maksymalnej szerokości wyświetlania. W poniższych tabelach wielkość tę zaznaczono literą *M*. Jeżeli jest ona opcjonalna, litera *M* znajduje się w nawiasach kwadratowych. Największa dopuszczalna wartość parametru *M* wynosi 255.

Wszystkie parametry opcjonalne są przedstawione w nawiasach kwadratowych.

Typy liczbowe

Typy liczbowe dzielą się na całkowitoliczbowe i zmiennoprzecinkowe. W przypadku tych ostatnich istnieje możliwość zadeklarowania liczby cyfr znaczących po przecinku — w tabelach wielkość ta jest oznaczona symbolem *D*. Maksymalna wartość parametru *D* jest wyznaczona przez mniejszą spośród dwóch liczb: 30 i *M* – 2 (tj. maksymalna szerokość wyświetlania minus 2 — jeden znak przecinka i jeden znak dla całkowitej części danej liczby).

W przypadku typów całkowitoliczbowych można je zawęzić do typu `UNSIGNED` tak, jak pokazano na listingu 8.1.

Dla wszystkich typów liczbowych można ustawić atrybut `ZEROFILL`. Powoduje on, że przy wyświetlaniu zawartości kolumn posiadających ten atrybut zapisane w nich liczby zostaną poprzedzone zerami. Jeżeli kolumna zostanie zdefiniowana jako `ZEROFILL`, automatycznie będzie ona również mieć typ `UNSIGNED`.

Typy całkowitoliczbowe zostały przedstawione w tabeli 8.5. W drugiej kolumnie tabeli zawarto dopuszczalny zakres danych w przypadku braku atrybutu `UNSIGNED` (pierwszy wiersz) i z ustawionym atrybutem `UNSIGNED` (drugi wiersz).

Tabela 8.5. *Typy całkowitoliczbowe*

Typ	Zakres	Wykorzystanie pamięci (w bajtach)	Opis
`TINYINT[(M)]`	–127..128 lub 0..255	1	Bardzo małe liczby całkowite
`BIT`			Synonim `TINYINT`
`BOOL`			Synonim `TINYINT`
`SMALLINT[(M)]`	–32768..32767 lub 0..65535	2	Małe liczby całkowite
`MEDIUMINT[(M)]`	–8388608..8388607 lub 0..16777215	3	Średnie liczby całkowite
`INT[(M)]`	$-2^{31}..2^{31}-1$ lub $0..2^{32}-1$	4	Zwykłe liczby całkowite
`INTEGER[(M)]`			Synonim typu `INT`
`BIGINT[(M)]`	$-2^{63}..2^{63}-1$ lub $0..2^{64}-1$	8	Duże liczby całkowite

Tabela 8.6 przedstawia typy zmiennoprzecinkowe.

Tabela 8.6. *Typy zmiennoprzecinkowe*

Typ	Zakres	Wykorzystanie pamięci (w bajtach)	Opis
`FLOAT(precyzja)`	zależnie od precyzji	różny	Używany do deklarowania liczb zmiennoprzecinkowych o pojedynczej lub podwójnej precyzji.
`FLOAT[(M, D)]`	±1.175494351E–38 ±3.402823466E+38	4	Liczby zmiennoprzecinkowe o pojedynczej precyzji. Typ ten jest równoznaczny z typem `FLOAT(4)`, pozwala przy tym na określenie szerokości wyświetlania i liczby cyfr znaczących po przecinku.
`DOUBLE[(M, D)]`	±1.7976931348623157E–308 ±2.2250738585072014E+308	8	Liczby zmiennoprzecinkowe o podwójnej precyzji. Typ ten jest równoznaczny z typem `FLOAT(8)`, pozwala przy tym na określenie szerokości wyświetlania i liczby cyfr znaczących po przecinku.
`DOUBLE PRECISION[(M, D)]`	jak wyżej		Synonim typu `DOUBLE[(M, D)]`
`REAL[(M,D)]`	jak wyżej		Synonim typu `DOUBLE[(M, D)]`
`DECIMAL[(M[, D])]`	różny	$M + 2$	Liczba zmiennoprzecinkowa przechowywana jako zmienna typu `CHAR`. Zakres zależy od wartości *M* — szerokości wyświetlania.
`NUMERIC[(M, D)]`	jak wyżej		Synonim typu `DECIMAL`
`DEC[(M, D)]`	jak wyżej		Synonim typu `DECIMAL`

Typy daty i czasu

MySQL obsługuje szereg typów daty i czasu — są one zaprezentowane w tabeli 8.7. Każdy z nich pozwala na wpisanie danych w formie liczbowej lub łańcucha znaków. Charakterystyczną cechą typu TIMESTAMP jest to, że jeżeli pole tego typu pozostanie niewypełnione, wówczas automatycznie zostaną w nim zapisane czas i data aktualnie wykonywanej operacji. Właściwość ta jest szczególnie przydatna do analizy transakcji.

Tabela 8.7. *Typy daty i czasu*

Typ	Zakres	Opis
DATE	1000-01-01 do 9999-12-31	Data wyświetlana w formacie RRRR-MM-DD.
TIME	–838:59:59 do 838:59:59	Czas wyświetlany w formacie GG:MM:SS. Zakres typu jest tak szeroki, że zapewne nigdy nie będzie w pełni wykorzystywany.
DATETIME	1000-01-01 00:00:00 do 9999-12-31 23:59:59	Data i czas wyświetlane w formacie RRRR-MM-DD GG:MM:SS.
TIMESTAMP[(*M*)]	1970-01-01 00:00:00 do roku 2037	Typ szczególnie przydatny do śledzenia transakcji. Format wyświetlania zależy od wartości parametru *M*, a górny zakres typu od systemu operacyjnego.
YEAR[(*2* | *4*)]	70 – 69 (czyli 1970 – 2069) lub 1901 – 2155	Rok wyświetlany w formie dwu- lub czterocyfrowej. Jak widać, każdy z nich ma odmienny zakres.

Tabela 8.8 prezentuje możliwe dostępne formaty wyświetlania wartości typu TIMESTAMP.

Tabela 8.8. *Formaty wyświetlania wartości typu TIMESTAMP*

Podany typ	Format wyświetlania
TIMESTAMP	RRRRMMDDGGMMSS
TIMESTAMP(14)	RRRRMMDDGGMMSS
TIMESTAMP(12)	RRMMDDGGMMSS
TIMESTAMP(10)	RRMMDDGGMM
TIMESTAMP(8)	RRRRMMDD
TIMESTAMP(6)	RRMMDD
TIMESTAMP(4)	RRMM
TIMESTAMP(2)	RR

Typy łańcuchowe

Typy łańcuchowe dzielą się na trzy grupy. Pierwsza z nich to klasyczne (zwykłe) łańcuchy znaków, czyli krótkie fragmenty tekstu. Do tej grupy zaliczamy typy CHAR (łańcuchy o stałej długości) oraz VARCHAR (łańcuchy o zmiennej długości). Dla każdego z nich można określić długość łańcucha. Łańcuchy zapisywane w kolumnach typu CHAR zostaną uzupełnione spacjami w celu pełnego wykorzystania dopuszczalnego limitu ich długości. Natomiast

w kolumnach typu VARCHAR łańcuchy są zapisywane w ich pierwotnej formie, dzięki czemu zajmują one tylko tyle pamięci, ile potrzeba (tak więc w przypadku wartości CHAR MySQL usunie końcowe znaki spacji przy ich *pobieraniu* z bazy, natomiast ewentualne znaki spacji znajdujące się na końcu łańcucha VARCHAR zostaną wyeliminowane w momencie *zapisania* go do bazy). Wybór któregoś z tych typów sprowadza się więc do rozstrzygnięcia dylematu między zwiększeniem szybkości działania kosztem używanej pamięci a ograniczeniem zużycia pamięci i spadkiem wydajności systemu. Zagadnienie to zostanie wyczerpująco omówione w rozdziale 11.

Do drugiej grupy należą typy TEXT i BLOB. Wartości obu wymagają zróżnicowanych ilości pamięci. Typy te są przeznaczone do przechowywania, odpowiednio, dłuższych tekstów oraz danych binarnych. Wartości typu BLOB to tzw. *duże obiekty binarne* (ang. *binary large objects*), stąd też wywodzi się jego nazwa. Obiekty te mogą przechowywać każdy rodzaj danych, np. obrazy, dźwięki itp.

W praktyce jedyna różnica pomiędzy oboma typami polega na tym, że w danych typu TEXT rozróżniana jest wielkość liter, typ BLOB natomiast tej właściwości nie posiada. Ponieważ oba typy pozwalają na zapisywanie dużych ilości danych, sposób ich wykorzystania jest nieco bardziej skomplikowany. Zagadnienie to zostanie szerzej przedstawione w rozdziale 11.

Trzecia grupa składa się z dwóch typów specjalnych SET i ENUM. Typ SET jest używany w celu zawężenia wartości danych, które mogą być zapisane w danej kolumnie, do z góry określonego zbioru wartości, przy czym dane zapisane w kolumnie mogą mieć więcej niż jedną wartość z tego zbioru. Podany zbiór wartości może zawierać co najwyżej 64 elementy.

ENUM to inaczej typ wyliczeniowy. Jest on bardzo podobny do typu SET, z jedną różnicą: kolumny typu ENUM mogą zawierać tylko jedną spośród określonego zbioru wartości lub wartość NULL, a zbiór wartości dopuszczalnych może zawierać do 65 535 elementów.

Tabele 8.9, 8.10 i 8.11 zawierają podsumowanie właściwości typów łańcuchowych. Tabela 8.9 przedstawia zwykłe typy łańcuchowe.

Tabela 8.9. *Zwykłe typy łańcuchowe*

Typ	Zakres	Opis
[NATIONAL] CHAR(*M*) [BINARY]	1 – 255 znaków	Łańcuch znaków o stałej długości *M*, gdzie *M* może przybierać wartości od 1 do 255. Słowo kluczowe NATIONAL wymusza użycie domyślnego zbioru znaków. Zbiór ten jest i tak domyślnie wykorzystywany przez MySQL, jednak opcja ta została udostępniona jako część standardu ANSI SQL. Słowo kluczowe BINARY wyłącza rozpoznawanie wielkości liter (domyślnie wielkość liter jest rozpoznawana).
CHAR	1	Synonim typu CHAR(1)
[NATIONAL] VARCHAR(*M*) [BINARY]	1 – 255 znaków	Łańcuch znaków o różnej długości, reszta jak wyżej.

Tabela 8.10 prezentuje typy TEXT i BLOB. Maksymalna liczba znaków w polu typu TEXT jest równa maksymalnej wielkości pliku, jaki można by przechowywać w tym polu, mierzonej w bajtach.

Tabela 8.10. *Typy TEXT i BLOB*

Typ	Maksymalna długość (w znakach)	Opis
TINYBLOB	$2^8 - 1$ (czyli 255)	Mały obiekt BLOB
TINYTEXT	$2^8 - 1$ (czyli 255)	Krótkie pole tekstowe
BLOB	$2^{16} - 1$ (czyli 65535)	Zwykły obiekt BLOB
TEXT	$2^1 - 1$ (czyli 65535)	Pole tekstowe o zwykłej długości
MEDIUMBLOB	$2^{24} - 1$ (czyli 16777215)	Średni obiekt BLOB
MEDIUMTEXT	$2^{24} - 1$ (czyli 16777215)	Pole tekstowe o średniej długości
LONGBLOB	$2^{32} - 1$ (czyli 4294967295)	Duży obiekt BLOB
LONGTEXT	$2^{32} - 1$ (czyli 4294967295)	Długie pole tekstowe

Tabela 8.11 prezentuje typy SET i ENUM.

Tabela 8.11. *Typy SET i ENUM*

Typ	Maksymalna ilość wartości w zbiorze	Opis
ENUM('*wartość1*','*wartość2*',...)	65535	W kolumnie tego typu może znajdować się tylko *jedna* wartość ze zbioru wartości dopuszczalnych lub NULL.
SET('*wartość1*','*wartość2*',...)	64	W kolumnie tego typu może znajdować się *podzbiór* zbioru wartości dopuszczalnych lub NULL.

Propozycje dalszych lektur

Więcej informacji na temat tworzenia bazy danych MySQL znajduje się w podręczniku elektronicznym, dostępnym pod adresem *http://www.mysql.com*.

W następnym rozdziale

Ponieważ znamy już metody rejestrowania użytkowników, zakładania baz danych i tworzenia tabel, możemy przejść do sposobów korzystania z bazy danych. W następnym rozdziale zostaną przedstawione metody zapisywania danych do tabel, ich modyfikacji i usuwania oraz wysyłania zapytań do bazy.