

» Idź do

- Spis treści
- Przykładowy rozdział

» Katalog książek

- Katalog online
- Zamów drukowany katalog

» Twój koszyk

- Dodaj do koszyka

» Cennik i informacje

- Zamów informacje o nowościach
- Zamów cennik

» Czytelnia

- Fragmenty książek online

» Kontakt

Helion SA
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl
© Helion 1991-2008

PHP, MySQL i Apache. Intensywny trening

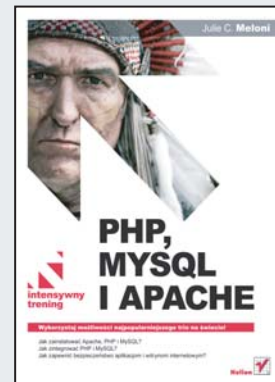
Autor: [Julie C. Meloni](#)

Tłumaczenie: Jarosław Dobrzański

ISBN: 978-83-246-2108-8

Tytuł oryginału: [Sams Teach Yourself PHP, MySQL and Apache All in One](#)

Format: 170×230, stron: 704



Wykorzystaj możliwości najpopularniejszego trio na świecie!

- Jak zainstalować Apache, PHP i MySQL?
- Jak zintegrować PHP i MySQL?
- Jak zapewnić bezpieczeństwo aplikacjom i witrynom internetowym?

PHP, MySQL i Apache – to informatyczne trio znają chyba wszyscy. W całej historii oprogramowania tak zgrane narzędzia spotykane są niezmiernie rzadko! Co sprawiło, że zdobyły one taką popularność? Odpowiedź jest krótka – cena, jakość i wydajność. Jeżeli dodać do tego gigantyczną wręcz społeczność, nic dziwnego, że jest to wiodące rozwiązanie, służące do tworzenia stron oraz aplikacji internetowych.

Dzięki tej książce dowiesz się, w jaki sposób rozpocząć pracę z tymi narzędziami oraz jak skonfigurować każde z nich. Po przygotowaniu środowiska pracy rozpoczniesz naukę PHP, tak aby pętle, tablice, funkcje i inne konstrukcje tego języka nie miały więcej dla Ciebie tajemnic. W kolejnych rozdziałach nauczysz się integrować PHP z MySQL. Poznasz tajniki procesu projektowania bazy danych, podstawowe polecenia SQL oraz sposoby operowania na danych z poziomu PHP. Ponadto zdobędziesz wiedzę na temat zaawansowanej konfiguracji serwera Apache, monitorowania jego pracy oraz sposobów wykorzystania protokołu SSL. Niewątpliwie ogromnym atutem książki są liczne przykłady, które pozwolą Ci szybko przyswoić wiedzę. Możesz również wykorzystać je bezpośrednio na Twoich stronach!

- Instalacja MySQL, Apache i PHP w systemach Linux/Unix, Windows, MacOS X
- Typy danych w PHP
- Operatory, tablice i funkcje w PHP
- Sterowanie przepływem
- Wykorzystanie obiektów w PHP
- Działania na ciągach znaków, czasie i datach
- Tworzenie formularzy
- Wykorzystanie cookies i sesji
- Integracja PHP i MySQL
- Projektowanie bazy danych
- Podstawowe polecenia MySQL
- Dostęp do danych zawartych w bazie z poziomu MySQL
- Wykorzystanie transakcji i procedur składowanych
- Przykładowe projekty: lista mailingowa, książka adresowa, forum dyskusyjne
- Ograniczanie dostępu do aplikacji
- Monitorowanie pracy serwera Apache
- Zaawansowana konfiguracja serwera Apache

Twórz atrakcyjne rozwiązania, oparte o PHP, MySQL i Apache!

Spis treści

Wprowadzenie	15
--------------	----

Część I Instalacja i konfiguracja

Rozdział 1. Szybka instalacja	21
-------------------------------	----

Instalacja w systemie Linux/Unix	21
Instalacja w systemie Windows	25
Instalacja w systemie Mac OS X	31
Rozwiązywanie problemów	35

Rozdział 2. Instalacja i konfiguracja MySQL	37
---	----

Wersja aktualna i przyszłe wersje MySQL	37
Jak zdobyć MySQL	38
Instalacja MySQL w systemie Linux/Unix	39
Instalacja MySQL w systemie Mac OS X	40
Instalacja MySQL w systemie Windows	42
Rozwiązywanie problemów instalacji	50
Podstawy bezpieczeństwa	50
Wprowadzenie do systemu uprawnień MySQL	52
Korzystanie z systemu uprawnień	54
Podsumowanie	57
Pytania i odpowiedzi	58
Warsztat	58

Rozdział 3. Instalacja i konfiguracja Apache	61
--	----

Wersja aktualna i przyszłe wersje Apache	61
Wybór sposobu instalacji	62
Instalacja Apache w systemie Linux/Unix	63
Instalacja Apache w systemie Mac OS X	66
Instalacja Apache w systemie Windows	66
Format pliku konfiguracyjnego Apache	69
Pliki dziennika Apache	75

Polecenia związane z Apache	76
Pierwsze uruchomienie Apache	78
Rozwiązywanie problemów	80
Podsumowanie	81
Pytania i odpowiedzi	82
Warsztat	82
Rozdział 4. Instalacja i konfiguracja PHP	85
Wersja aktualna i przyszłe wersje PHP	85
Kompilacja PHP w systemie Linux/Unix	86
Instalacja PHP w systemie Mac OS X	89
Instalacja PHP w systemie Windows	91
Plik php.ini	94
Testowanie	94
Gdzie znaleźć pomoc	95
Podstawy skryptów PHP	97
Podsumowanie	102
Pytania i odpowiedzi	103
Warsztat	103
 Część II Struktura języka PHP	
Rozdział 5. Podstawowe elementy języka PHP	107
Zmienne	107
Typy danych	110
Operatory i wyrażenia	117
Stałe	126
Podsumowanie	127
Pytania i odpowiedzi	128
Warsztat	128
 Rozdział 6. Sterowanie przepływem w PHP	131
Zmiana przepływu	132
Pętle	138
Bloki kodu PHP	146
Podsumowanie	149
Pytania i odpowiedzi	149
Warsztat	150

Rozdział 7. Funkcje	153
Czym jest funkcja?	153
Wywoływanie funkcji	154
Definiowanie funkcji	156
Zwracanie wartości przez funkcje użytkownika	158
Zasięg zmiennych	159
Przechowywanie wartości pomiędzy wywołaniami funkcji... ..	163
Więcej o argumentach	166
Sprawdzanie istnienia funkcji	169
Podsumowanie	171
Pytania i odpowiedzi	172
Warsztat	172
Rozdział 8. Tablice	175
Czym jest tablica?	175
Tworzenie tablic	176
Niektóre funkcje operujące na tablicach	180
Podsumowanie	182
Pytania i odpowiedzi	182
Warsztat	182
Rozdział 9. Obiekty	185
Tworzenie obiektu	185
Dziedziczenie	191
Podsumowanie	193
Pytania i odpowiedzi	193
Warsztat	194
 Część III Pierwsze kroki z kodem	
Rozdział 10. Ciągi znaków, data i czas	197
Formatowanie ciągów znaków	198
Analizowanie ciągów znaków	208
Operacje na ciągach znaków	213
Funkcje operujące na dacie i czasie	220
Inne funkcje operujące na ciągach znaków, datach i czasie	226
Podsumowanie	227
Warsztat	227

Rozdział 11. Formularze	231
Tworzenie prostego formularza	231
Przekazywanie informacji w tablicach	234
Łączenie kodu HTML i PHP w jednym skrypcie	237
Zapisywanie informacji o stanie w ukrytym polu	239
Przekierowania	241
Wysyłanie poczty elektronicznej	243
Przesyłanie plików	249
Podsumowanie	253
Warsztat	254
Rozdział 12. Cookies i sesje	257
Wprowadzenie do cookies	257
Tworzenie cookie	259
Usuwanie cookie	261
Wprowadzenie do sesji	262
Otwieranie sesji	262
Zmienne sesyjne	263
Przekazywanie identyfikatora sesji w adresie	268
Niszczanie sesji i usuwanie zmiennych	269
Zastosowania sesji	269
Podsumowanie	270
Pytania i odpowiedzi	271
Warsztat	272
Rozdział 13. Pliki i katalogi	273
Dołączanie plików za pomocą funkcji include()	274
Weryfikacja plików	279
Tworzenie i usuwanie plików	283
Otwieranie plików do zapisu, odczytu i dopisywania	284
Odczytywanie danych z pliku	285
Zapisywanie i dopisywanie danych do pliku	290
Operacje na katalogach	293
Otwieranie potoków do i z procesów za pomocą funkcji popen()	296
Uruchamianie poleceń za pomocą funkcji exec()	298
Uruchamianie poleceń funkcjami system() i passthru()	300
Podsumowanie	302
Pytania i odpowiedzi	303
Warsztat	303

Rozdział 14. Obrazki	307
Proces powstawania obrazka	307
Konieczne zmiany w PHP	308
Rysowanie nowego obrazka	309
Rysowanie wykresów	313
Modyfikacja istniejących obrazków	317
Tworzenie obrazków na podstawie danych przesłanych przez użytkownika	321
Zastosowanie obrazków tworzonych skryptami	324
Podsumowanie	328
Pytania i odpowiedzi	329
Warsztat	329

Część IV Integracja PHP i MySQL

Rozdział 15. Tajniki procesu projektowania bazy danych	333
Rola dobrego projektu bazy danych	333
Typy relacji między tabelami	334
Normalizacja	338
Postępowanie zgodnie z procesem projektowania	341
Podsumowanie	343
Pytania i odpowiedzi	343
Warsztat	343

Rozdział 16. Podstawowe polecenia SQL	345
Typy danych w MySQL	346
Składnia tworzenia tabel	350
Używanie polecenia INSERT	351
Stosowanie polecenia SELECT	353
Używanie WHERE w zapytaniach	356
Selekcja z kilku tabel	358
Modyfikowanie rekordów za pomocą polecenia UPDATE	364
Używanie polecenia REPLACE	368
Stosowanie polecenia DELETE	369
Często stosowane funkcje MySQL operujące na ciągach tekstowych	371
Korzystanie z funkcji daty i czasu w MySQL	379
Podsumowanie	393
Pytania i odpowiedzi	395
Warsztat	396

Rozdział 17. Transakcje i procedury składowane w MySQL	399
Czym są transakcje?	399
Czym są procedury składowane?	403
Podsumowanie	405
Pytania i odpowiedzi	405
Warsztat	406
Rozdział 18. Interakcja z MySQL z poziomu PHP	407
Funkcje MySQL a funkcje MySQLi	407
Łączenie się z MySQL poprzez PHP	407
Operowanie na danych z bazy MySQL	411
Podsumowanie	419
Pytania i odpowiedzi	419
Warsztat	420
 Część V Proste projekty	
Rozdział 19. Zarządzanie prostą listą mailingową	423
Opracowywanie mechanizmu subskrypcji	423
Budowa mechanizmu mailingu	433
Podsumowanie	436
Pytania i odpowiedzi	437
Warsztat	437
Rozdział 20. Tworzenie internetowej książki adresowej	439
Planowanie i tworzenie tabel w bazie danych	439
Tworzenie pliku dołączanego ze wspólnymi funkcjami	442
Tworzenie menu	443
Tworzenie mechanizmu dodawania rekordów	444
Przeglądanie rekordów	450
Tworzenie mechanizmu usuwania rekordów	458
Uzupełnianie istniejących rekordów	460
Podsumowanie	467
Pytania i odpowiedzi	468
Warsztat	468
Rozdział 21. Tworzenie prostego forum dyskusyjnego	471
Projektowanie tabel w bazie danych	471
Tworzenie pliku dołączanego ze wspólnymi funkcjami	472

Tworzenie formularzy wprowadzania danych i skryptów	473
Wyświetlanie listy tematów	477
Wyświetlanie postów w temacie	481
Dodawanie postu w wybranym temacie	485
Podsumowanie	489
Pytania i odpowiedzi	490
Warsztat	491
Rozdział 22. Tworzenie witryny sklepu internetowego	493
Planowanie i tworzenie tabel w bazie danych	493
Wyświetlanie kategorii artykułów	499
Wyświetlanie artykułów	503
Podsumowanie	506
Warsztat	506
Pytania i odpowiedzi	507
Rozdział 23. Tworzenie mechanizmu koszyka z zakupami	509
Planowanie i tworzenie tabel	509
Integracja koszyka z witryną sklepową	512
Sposoby dokonywania płatności i sekwencja kasowa	521
Podsumowanie	524
Warsztat	524
Rozdział 24. Tworzenie prostego kalendarza	525
Tworzenie prostego kalendarza wyświetlanego na ekranie	525
Tworzenie biblioteki kalendarza	541
Podsumowanie	549
Pytania i odpowiedzi	550
Warsztat	550
Rozdział 25. Ograniczanie dostępu do aplikacji	551
Istota uwierzytelniania	551
Możliwości funkcjonalne modułu uwierzytelniającego serwera Apache ...	554
Apache jako narzędzie kontroli dostępu	559
Wiązane zastosowanie metod kontroli dostępu	562
Ograniczenie dostępu na podstawie metod HTTP	563
Ograniczenie dostępu na podstawie wartości cookies	564
Podsumowanie	570
Pytania i odpowiedzi	571
Warsztat	572

Rozdział 26. Monitorowanie i prowadzenie dzienników aktywności serwera	573
Standardowe odnotowywanie dostępu do serwera	573
Standardowy tryb odnotowywania błędów serwera Apache	580
Zarządzanie dziennikami serwera Apache	582
Odnotowywanie informacji w bazie danych	585
Podsumowanie	590
Pytania i odpowiedzi	590
Warsztat	590
Rozdział 27. Lokalizacja aplikacji	593
Internacjonalizacja i lokalizacja	593
Zestawy znaków	594
Modyfikacje środowiska	596
Tworzenie zlokalizowanej struktury strony	597
Podsumowanie	604
Pytania i odpowiedzi	604
Warsztat	605
Rozdział 28. Korzystanie z XML	607
Co to jest XML?	607
Dostęp do dokumentów XML z poziomu PHP za pomocą funkcji modelu DOM	610
Dostęp do danych XML z poziomu PHP za pomocą funkcji SimpleXML ...	613
Podsumowanie	617
Pytania i odpowiedzi	617
Warsztat	618
 Część VI Administrowanie i dostrajanie	
Rozdział 29. Poprawianie wydajności i wirtualny hosting na serwerze Apache	621
Kwestie skalowalności	622
Testowanie serwera pod obciążeniem przy użyciu ApacheBench	626
Aktywne dostrajanie wydajności	629
Zapobieganie nadużyciom	631
Implementacja wirtualnego hostingu	632
Podsumowanie	637
Pytania i odpowiedzi	638
Warsztat	639

Rozdział 30. Bezpieczny serwer WWW	641
Potrzeba bezpieczeństwa	641
Protokół SSL	642
Uzyskiwanie i instalacja narzędzi SSL	648
Zarządzanie certyfikatami	651
Konfiguracja SSL	654
Podsumowanie	655
Pytania i odpowiedzi	656
Warsztat	656
Rozdział 31. Optymalizacja i dostrajanie MySQL	657
Tworzenie zoptymalizowanej platformy	658
Opcje inicjalizacyjne MySQL	659
Optymalizacja struktury tabel	662
Optymalizacja zapytań	662
Korzystanie z polecenia FLUSH	664
Korzystanie z polecenia SHOW	665
Podsumowanie	671
Pytania i odpowiedzi	672
Warsztat	672
Rozdział 32. Aktualizacja oprogramowania	675
Trzymanie ręki na pulsie	675
Aktualizacja MySQL	677
Aktualizacja Apache	678
Aktualizacja PHP	679
Podsumowanie	680
Warsztat	681
Skorowidz	683

Rozdział 15

Tajniki procesu projektowania bazy danych

W tym rozdziale poznamy tok rozumowania prowadzący do stworzenia relacyjnej bazy danych. Po tym skoncentrowanym na teorii rozdziale od razu zagłębimy się w naukę podstawowych poleceń MySQL w ramach przygotowań do zintegrowania bazy MySQL z naszymi aplikacjami.

Zagadnienia omówione w tym rozdziale to:

- ▶ Niektóre zalety dobrego projektu bazy danych
- ▶ Trzy typy relacji między tabelami
- ▶ Jak znormalizować bazę danych
- ▶ Jak wdrożyć proces właściwego projektowania baz danych

Rola dobrego projektu bazy danych

Dobry projekt bazy danych jest kluczowym składnikiem wydajnej aplikacji, tak jak opływowa karoseria jest ważną częścią samochodu wyścigowego. Jeżeli samochód nie ma opływowych kształtów, będzie stawiał opór i wolniej jechał. Jeżeli nie zoptymalizujemy relacji, nasza baza nie będzie działać tak wydajnie, jak by mogła. Myślenie o relacjach i wydajności bazy jest elementem *normalizacji*.

Uwaga

Normalizacja to proces tworzenia takiej struktury danych, w której zminimalizowano powtarzanie i niespójności.

Poza wydajnością istnieje jeszcze kwestia utrzymania — nasza baza powinna być łatwa w utrzymaniu. Sprowadza się to do przechowywania jak najmniejszej ilości powtarzanych danych (albo całkowitej eliminacji powtórzeń). Jeżeli mamy dużo powtarzających się danych i nagle zachodzi zmiana w jednym z wystąpień takich danych (na przykład zmiana nazwiska), to trzeba zmienić wszystkie wystąpienia tej danej. Aby wyeliminować powtarzanie i ułatwić utrzymanie danych, można stworzyć tabelę możliwych wartości i odnosić się do tych wartości poprzez klucz. W ten sposób, jeżeli wartość ulegnie zmianie, to zmianę tę wystarczy wprowadzić tylko raz — w głównej tabeli. Odwołania występujące w innych tabelach pozostają bez zmian.

Przypuśćmy, że jesteśmy odpowiedzialni za utrzymanie bazy danych studentów oraz przedmiotów, na które się zapisali. Jeżeli 35 studentów zapisało się na ten sam przedmiot, niech to będzie „wyższa matematyka”, to nazwa tego przedmiotu wystąpi w tabeli 35 razy. Jeżeli wykładowca stwierdzi, że trzeba zmienić nazwę przedmiotu na „matematyka zaawansowana”, to musimy wprowadzić zmiany w 35 rekordach. Gdyby baza zbudowana została tak, że nazwy przedmiotów występowałyby w jednej tabeli, a w rekordach studentów występowałyby tylko identyfikator przedmiotu, to aby zmienić nazwę przedmiotu, wystarczyłoby dokonać modyfikacji w jednym rekordzie, a nie w 35.

Korzyści z posiadania dobrze zaplanowanej i zaprojektowanej bazy danych są niezliczone. Im więcej pracy włożymy w bazę na początku, tym mniej będziemy mieli do zrobienia później. Przeprojektowywanie bazy danych po publicznym wdrożeniu aplikacji nie jest zbyt dobrym pomysłem (choć to się zdarza), a rezultaty są kosztowne.

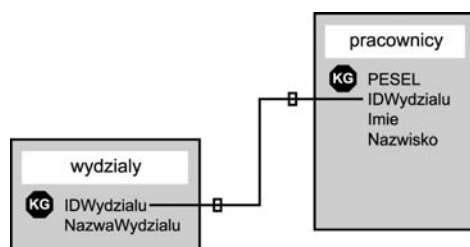
Tak więc zanim w ogóle zabierzemy się do pisania kodu aplikacji, poświęćmy stosowny czas na zaprojektowanie bazy danych. Reszta tego rozdziału poświęcona jest relacjom i normalizacji — dwóm ważnym elementom bazodanowej układanki.

Typy relacji między tabelami

Relacje między tabelami można podzielić na:

- ▶ Relacje jeden do jednego
- ▶ Relacje jeden do wielu
- ▶ Relacje wiele do wielu

Przypuśćmy, że mamy tabelę o nazwie *pracownicy*, która zawiera numer PESEL każdej osoby, jej nazwisko oraz wydział, w którym pracuje. Powiedzmy, że mamy też odrębną tabelę zwaną *wydziały*, która zawiera listę wszystkich wydziałów składającą się z identyfikatora wydziału i jego nazwy. Pole zawierające identyfikator wydziału w tabeli *pracownicy* odpowiada identyfikatorowi z tabeli *wydziały*. Tego rodzaju relację przedstawia rysunek 15.1. Oznaczenie KG tuż przy nazwie pola oznacza klucz główny.

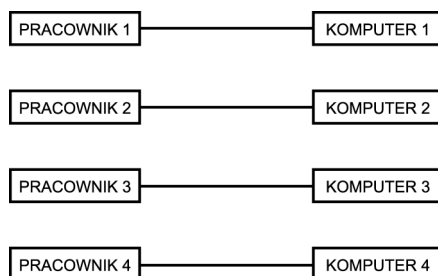


RYСУNEK 15.1.
Tabele „pracownicy” i „wydziały” powiązane poprzez klucz „IDWydziału”

W kolejnych punktach opisano dokładniej każdy z typów relacji.

Relacje jeden do jednego

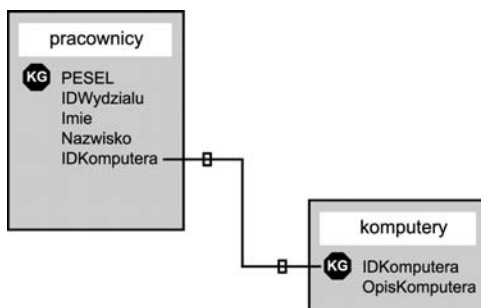
W relacji jeden do jednego klucz pojawia się tylko raz w powiązanej tabeli. Tabel *pracownicy* i *wydziały* nie łączy relacja jeden do jednego, ponieważ do jednego wydziału niewątpliwie należy wielu pracowników. Relacja jeden do jednego istnieje na przykład wtedy, gdy każdemu pracownikowi jest przypisany jeden komputer. Taką relację pokazuje rysunek 15.2.



RYСУNEK 15.2.
Każdemu pracownikowi przypisany jest jeden komputer

Tabele *pracownicy* i *komputery* w naszej bazie wyglądałyby tak, jak przedstawia to rysunek 15.3, który reprezentuje relację jeden do jednego.

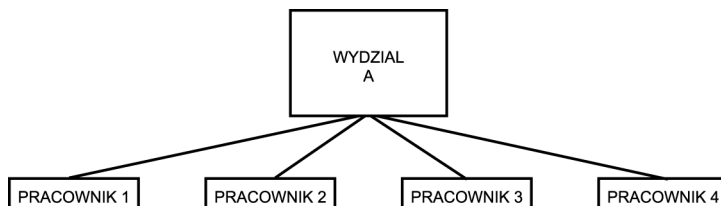
RYSUNEK 15.3.
Relacja jeden
do jednego
w modelu
danych



Relacje jeden do wielu

W relacji jeden do wielu klucze z jednej tabeli występują wielokrotnie w tabeli powiązanej. Relację jeden do wielu ilustruje przykład z rysunku 15.1, gdzie powiązani są pracownicy z wydziałami. Praktycznym przykładem takiej relacji jest struktura organizacyjna wydziałów (rysunek 15.4).

RYSUNEK 15.4.
Jeden wydział
zawiera wielu
pracowników



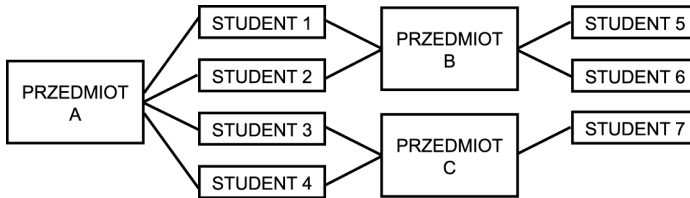
Relacja jeden do wielu jest relacją najczęściej występującą. Innym praktycznym przykładem jest zastosowanie skrótów nazw krajów. Każdy kraj na świecie ma swój niepowtarzalny identyfikator (Polska ma PL, Irlandia IE itd.).

Jeżeli mamy ośmiu klientów w Polsce i pięciu w Irlandii, to w tabeli użyjemy jedynie dwóch różnych skrótów. Jeden skrót (PL) reprezentuje relację jeden do ośmiu, a drugi (IE), relację jeden do pięciu.

Relacje wiele do wielu

Relacje wiele do wielu często są przyczyną problemów w praktycznych przykładach znormalizowanych baz danych — powszechne jest rozbijanie relacji wiele do wielu na kilka relacji jeden do wielu. W relacjach wiele do wielu wartość klucza z jednej tabeli może występować wielokrotnie w powiązanej tabeli. Na razie brzmi to jak opis relacji jeden do wielu. Sęk w tym, że działa to w obie strony, co oznacza, że klucz główny z drugiej tabeli może również wielokrotnie występować w pierwszej tabeli.

Relację tego typu spróbujemy wyjaśnić, posługując się wcześniejszym przykładem ze studentami i przedmiotami. Student ma identyfikator i nazwisko. Przedmiot ma identyfikator i nazwę. Student zwykle zapisuje się na więcej niż jeden przedmiot, a na dany przedmiot jest zapisanych więcej studentów niż jeden, co widać na rysunku 15.5.



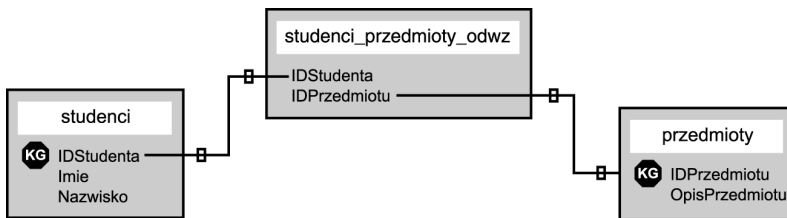
RYSunEK 15.5.
Student zapisuje się na przedmioty, a na przedmiot zapisani są studenci

Jak widać, tego rodzaju relacja nie jest łatwą metodą kojarzenia tabel. Nasze tabele wyglądają tak jak na rysunku 15.6 — na pozór nie są ze sobą w relacji.



RYSunEK 15.6.
Tabela studentów i tabela przedmiotów, niepowiązane z sobą

Aby stworzyć teoretyczną relację wiele do wielu, konieczne byłoby stworzenie tabeli przejściowej, która stanowi pomost między dwiema tabelami i opisuje ich wzajemne odwzorowania. Może ona wyglądać tak jak na rysunku 15.7.



RYSunEK 15.7.
Tabela studenci_przedmioty_odwz służy jako element pośredniczący

Jeżeli wykorzystamy informacje z rysunku 15.5 i wstawimy je do tabeli przejściowej, otrzymamy tabelę widoczną na rysunku 15.8.

Jak widać, wielu studentów i wiele przedmiotów może z powodzeniem współistnieć z sobą w tabeli studenci_przedmioty_odwz.

Po takim wprowadzeniu do typów relacji normalizacja powinna być błahostką.

RYSUNEK 15.8.
Tabela studenci_
przedmioty_odw
z wypełniona
danymi

IDSTUDENTA	IDPRZEDMIOTU
STUDENT 1	PRZEDMIOT A
STUDENT 2	PRZEDMIOT A
STUDENT 3	PRZEDMIOT A
STUDENT 4	PRZEDMIOT A
STUDENT 5	PRZEDMIOT B
STUDENT 6	PRZEDMIOT B
STUDENT 7	PRZEDMIOT C
STUDENT 1	PRZEDMIOT B
STUDENT 2	PRZEDMIOT B
STUDENT 3	PRZEDMIOT C
STUDENT 4	PRZEDMIOT C

Normalizacja

Normalizacja jest po prostu zbiorem reguł, które ostatecznie mają ułatwić nam życie, jeżeli pełnimy funkcję administratora bazy. Jest to sztuka organizacji bazy danych w taki sposób, by tabele były powiązane tam, gdzie jest to stosowne, i by była możliwość ich łatwej rozbudowy.

Zbiory reguł stosowane w normalizacji są nazywane *postaciami normalnymi*. Jeżeli nasz projekt bazy jest zgodny z pierwszym zbiorem reguł, jest uważany za *pierwszą postać normalną* bazy. Jeżeli nasz projekt jest zgodny z pierwszymi trzema zbiorami reguł normalizacyjnych, to bazę możemy uznać za *trzecią postać normalną*.

W tym rozdziale poznamy wszystkie reguły pierwszej, drugiej i trzeciej postaci normalnej, aby móc się ich trzymać przy tworzeniu własnych aplikacji. Zastosujemy przykładowy zbiór tabel z bazy studentów i przedmiotów i doprowadzimy go do trzeciej postaci normalnej.

Problemy z tabelą prostą

Zanim przejdziemy do pierwszej postaci normalnej, musimy wyjść od czegoś, co można poddać normalizacji. W przypadku baz danych jest to *tabela prosta*. Tabela prosta jest jak arkusz kalkulacyjny — ma nieokreśloną liczbę kolumn. Nie istnieją relacje między odrębnymi tabelami — wszystkie potrzebne dane zostały zgromadzone w jednej dużej tabeli. Takie rozwiązanie jest nieefektywne i zajmuje więcej miejsca na dysku niż baza znormalizowana.

Przyjmijmy, że w bazie studentów i przedmiotów występują następujące pola:

- ▶ **NazwiskoStudenta** — imię i nazwisko studenta.
- ▶ **IDprzedmiotu1** — identyfikator pierwszego przedmiotu wybranego przez studenta.
- ▶ **OpisPrzedmiotu1** — opis pierwszego przedmiotu wybranego przez studenta.
- ▶ **WykladowcaPrzedmiotu1** — wykladowca pierwszego przedmiotu wybranego przez studenta.
- ▶ **IDprzedmiotu2** — identyfikator drugiego przedmiotu wybranego przez studenta.
- ▶ **OpisPrzedmiotu2** — opis drugiego przedmiotu wybranego przez studenta.
- ▶ **WykladowcaPrzedmiotu2** — wykladowca drugiego przedmiotu wybranego przez studenta.
- ▶ Kolumny **IDPrzedmiotu**, **OpisPrzedmiotu** i **WykladowcaPrzedmiotu** powtarzają się wielokrotnie, aż ujęte zostaną wszystkie przedmioty wybrane przez studenta w trakcie jego studiów.

Wiedząc to, co już wiemy, powinniśmy być w stanie zidentyfikować pierwszy obszar problemów: kolumny **IDPrzedmiotu**, **OpisPrzedmiotu** i **WykladowcaPrzedmiotu** to powtarzające się grupy.

Eliminacja powtórzeń to pierwszy krok normalizacji, więc za chwilę doprowadzimy naszą tabelę do pierwszej postaci normalnej. Gdyby tabela pozostała w swojej płaskiej postaci, otrzymalibyśmy wiele pustej przestrzeni i wiele przestrzeni zajętej niepotrzebnie — nie jest to efektywna struktura tabel.

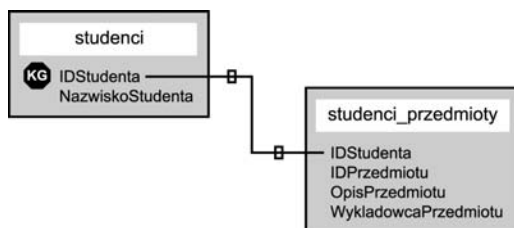
Pierwsza postać normalna

Reguły pierwszej postaci normalnej są następujące:

- ▶ Wyeliminować powtarzające się informacje.
- ▶ Stworzyć odrębne tabele dla powiązanych z sobą danych.

Zastanówmy się nad strukturą tabeli prostej z wieloma powtórzonymi grupami kolumn z bazy danych studentów i przedmiotów; może w niej zidentyfikować dwa odrębne tematy: studenci i zajęcia? Doprowadzenie naszej bazy do pierwszej postaci normalnej wymagałoby stworzenia dwóch tabel: jednej dla studentów, a drugiej dla przedmiotów, co widać na rysunku 15.9.

RYSUNEK 15.9.
Podział
tabeli prostej
na dwie tabele



Dwie otrzymane tabele reprezentują teraz relację jeden do wielu jednego studenta z wieloma przedmiotami. Studenci mogą wybierać tyle przedmiotów, ile chcą, i nie są ograniczeni liczbą grup kolumn `IDPrzedmiotu`, `OpisPrzedmiotu` i `WykladowcaPrzedmiotu` występujących w tabeli prostej.

Następny krok to doprowadzenie tabel do drugiej postaci normalnej.

Druga postać normalna

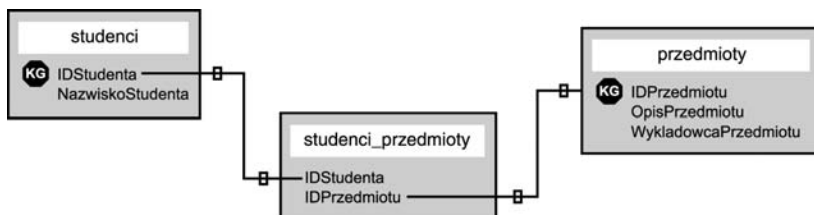
Zasada drugiej postaci normalnej brzmi tak:

- ▶ Żaden atrybut niebędący kluczem nie zależy od części klucza głównego.

Mówiąc prościej — oznacza to, że jeżeli pola naszej tabeli nie są w całości powiązane z kluczem głównym, to mamy jeszcze coś do zrobienia. W przykładzie ze studentami i przedmiotami musimy zebrać przedmioty w oddzielnej tabeli i zmodyfikować tabelę `studenci_przedmioty`.

Kolumny `IDPrzedmiotu`, `OpisPrzedmiotu` i `WykladowcaPrzedmiotu` mogą stać się tabelą zwaną `przedmioty` z kolumną `IDprzedmiotu` w roli klucza głównego. Tabela `studenci_przedmioty` powinna wówczas zawierać tylko dwa pola: `IDStudenta` i `IDPrzedmiotu`. Nową strukturę przedstawiono na rysunku 15.10.

RYSUNEK 15.10.
Doprowadzenie
tabel do drugiej
postaci
normalnej



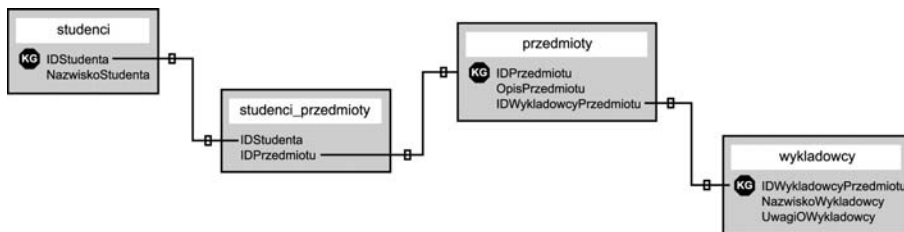
Struktura ta powinna wyglądać znajomo, jako relacja wiele do wielu z zastosowaniem pośredniczącej tabeli odwzorowującej. Trzecia postać normalna to ostatnia postać, jaka nas interesuje. Jak się zaraz okaże, zasada nią rządząca jest równie prosta, jak w przypadku pierwszych dwóch postaci.

Trzecia postać normalna

Zasada trzeciej postaci normalnej brzmi:

- ▶ Żaden atrybut nie zależy od innych atrybutów, które nie są kluczami.

Reguła ta oznacza po prostu, że musimy przyjrzeć się naszym tabelom i sprawdzić, czy istnieją jeszcze jakieś pola, które można rozbić i które nie będą zależne od klucza. Szukajmy powtarzających się danych, a szybko znajdziemy odpowiedź — wykładowcy. Jeden wykładowca z pewnością będzie wyładał więcej niż jeden przedmiot. Jednakże `WykładowcaPrzedmiotu` nie jest kluczem żadnego typu. Jeżeli więc rozbijemy te dane i stworzymy odrębną tabelę tylko po to, by zwiększyć efektywność i ułatwić utrzymanie bazy, to otrzymamy trzecią postać normalną (rysunek 15.11).



RYСУNEK 15.11.
Doprowadzanie
tabel do trzeciej
postaci
normalnej

Trzecia postać normalna zwykle wystarczy, by usunąć powtórzenia i zapewnić elastyczność oraz łatwość rozbudowy. W następnym podrozdziale otrzymamy kilka wskazówek co do procesu myślowego prowadzącego do zaprojektowania struktury bazy oraz jego powiązania z procesem tworzenia całej aplikacji.

Postępowanie zgodnie z procesem projektowania

Największym problemem przy projektowaniu aplikacji jest brak jej wcześniejszego przemyślenia. W odniesieniu do aplikacji bazodanowych proces projektowania musi uwzględniać wnikliwą analizę bazy danych — co powinna przechowywać, jakie relacje występują między danymi i, co najważniejsze, czy jest skalowalna.

Ogólne kroki procesu projektowania to:

- ▶ Definiowanie celów.
- ▶ Projektowanie struktur danych (tabele, pola).
- ▶ Rozpoznanie relacji.

- ▶ Zdefiniowanie i implementacja reguł obszaru zastosowania.
- ▶ Stworzenie aplikacji.

Tworzenie aplikacji jest krokiem ostatnim, a nie pierwszym! Wielu programistów wymyśla aplikację, po czym pisze ją, a następnie próbuje wpasować w nią zbiór pól bazy danych. Takie podejście to zaczynanie od końca, jest nieefektywne i będzie nas kosztować czas oraz pieniądze.

Zanim zaczniemy jakiegokolwiek proces projektowania aplikacji, warto usiąść i go przedyskutować. Jeżeli nie potrafimy opisać tworzonej aplikacji pod kątem celów, odbiorców i rynku docelowego, to znaczy, że nie jesteśmy gotowi do jej budowy, nie mówiąc już o modelowaniu bazy danych.

Po opisanii innym, co będzie robić nasza aplikacja, i uzyskaniu ich aprobaty możemy zacząć myśleć o tabelach, jakie chcemy stworzyć. Zacznijmy od wielkich tabel prostych, ponieważ kiedy już je narysujemy, będziemy mogli zastosować właśnie zdobyte umiejętności normalizacyjne. W ten sposób znajdziemy powtórzenia i zwizualizujemy relacje.

Następnym krokiem jest normalizacja. Przejdźmy od tabeli prostej do pierwszej postaci normalnej, i po kolei aż do trzeciej, jeżeli to możliwe. Posługujmy się kartkami papieru, ołówkiem, fiszkami samoprzylepnymi i czymkolwiek, co pomoże zwizualizować tabele i relacje między nimi. To żaden wstyd modelować dane z pomocą fiszek, zanim będziemy gotowi do stworzenia samych tabel. Poza tym jest to o wiele tańsze niż kupowanie oprogramowania, które zrobi to za nas. Programy modelujące kosztują od pięciuset do kilku tysięcy złotych!

Kiedy już mamy wstępny model danych, spójrzmy na niego z perspektywy aplikacji. Albo spójrzmy na niego z perspektywy osoby, która będzie z aplikacji korzystać. W tym miejscu definiujemy reguły obszaru zastosowania i sprawdzamy, czy nasz model się sprawdza. Przykładem takiej reguły dla aplikacji rejestracji internetowej może być „każdy użytkownik musi mieć jeden adres e-mail, który nie może należeć do żadnego innego użytkownika”. Jeżeli pole AdresEmail nie było polem z niepowtarzalnymi wartościami w naszym modelu, znaczy to, że model nie spełnia reguł obszaru zastosowania.

Dopiero po zastosowaniu wszystkich reguł obszaru zastosowania na naszym modelu danych może się rozpocząć programowanie aplikacji. Możemy być spokojni o to, że nasz model danych jest spójny i że programując, nie zalamujemy się w kącie pokoju. A to zdarza się dość często.

Podsumowanie

Trzymanie się prawidłowego procesu projektowania bazy danych to jedyny sposób na stworzenie efektywnej, elastycznej i łatwej do utrzymania aplikacji. Ważnym aspektem projektowania bazy danych jest wykorzystanie relacji między tabelami zamiast wrzucania wszystkich danych do jednej prostej tabeli. Relacje mogą być jeden do jednego, jeden do wielu i wiele do wielu.

Stosowanie relacji w celu prawidłowej organizacji danych jest nazywane *normalizacją*. Istnieje wiele poziomów normalizacji, ale podstawowe to pierwsza, druga i trzecia postać normalna. Z każdym poziomem wiąże się reguła lub reguły, którym trzeba sprostać. Trzymanie się tych reguł pomaga w stworzeniu dobrze zorganizowanej i elastycznej bazy danych.

Aby przeprowadzić pomysł od jego narodzin do realizacji, należy trzymać się procesu projektowania. Proces ten sprowadza się do tego, by pomyśleć, zanim zaczniemy działać. Przedyskutujmy reguły, wymogi i cele, a dopiero potem stworzymy ostateczną wersję znormalizowanych tabel.

Pytania i odpowiedzi

P: *Czy istnieją tylko trzy postacie normalne?*

O: Nie. Postaci normalnych jest więcej. Dodatkowe postacie to postać normalna Boyce-Codda, czwarta postać normalna, piąta postać normalna, zwana też postacią chroniącą złączenia. Dość rzadko doprowadza się bazę do tych postaci, ponieważ koszty pracy i utrata efektywności przeważają nad ewentualnymi korzyściami.

Warsztat

Warsztaty mają na celu utrwalenie i sprawdzenie zdobytej wiedzy, powinny też pokazać, jak zastosować ją w praktyce.

Test

1. Wymień trzy typy relacji między danymi.
2. Jak należy radzić sobie z trudnościami w reprezentacji relacji wiele do wielu w efektywnej bazie danych?

Odpowiedzi

1. Jeden do jednego, jeden do wielu, wiele do wielu.
2. Zbudować szereg relacji jeden do wielu poprzez stworzenie pośredniczącej tabeli odwzorowującej.

Ćwiczenie

Objaśnij każdą z trzech postaci normalnych osobie, która pracuje z arkuszami kalkulacyjnymi i tabelami prostymi.