helion.pl
KSIĘGARNIA INTERNETOWA · WYDAWNICTWO

IDŹ DO
PRZYKŁADOWY ROZDZIAŁ
SPIS TREŚCI

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

Relacyjne bazy danych



Autorzy: Mark Whitehorn, Bill Marklyn
Tłumaczenie: Marek Pętlicki
ISBN: 83-7361-095-2
Tytuł oryginału: Inside Relational Databases 2nd Edition
Format: B5, stron: 338

Relacyjne bazy danych stanowią podstawę większości współczesnych systemów informatycznych. Choć poszczególne systemy zarządzania bazami danych różnią się między sobą w wielu aspektach, są jednak oparte na wspólnych podstawach teoretycznych. Jeśli zrozumiesz ten wspólny fundament, będziesz mógł z łatwością budować na nim własne aplikacje, niezależnie od tego, czy jako systemu bazodanowego użyjesz komercyjnego Oracle'a lub MS SQL-a, czy też bezpłatnego PostgreSQL.

Książka „Relacyjne bazy danych" została napisana w celu jak najbardziej przystępnego objaśnienia zagadnień relacyjnego modelu danych oraz jego znaczenia dla projektantów i twórców baz danych. Objaśnienia tych, często skomplikowanych zagadnień, przybliżają tajniki relacyjnego modelu danych wykorzystując przykłady, a nie wzory matematyczne. Dzięki ich zrozumieniu będziesz mógł projektować bazy danych szybsze, bardziej elastyczne i lepiej dopasowane do zadań, jakie mają realizować.

Poznasz:

- Podstawowe pojęcia związane z bazami danych: tabele, rekordy, pola
- Sposoby pobierania danych za pomocą zapytań
- Tworzenie raportów z wyselekcjonowanych danych
- Projektowanie baz z uwzględnieniem związków między danymi, klucze i indeksy
- Sposoby przestrzegania reguł integralności danych
- Tworzenie zaawansowanych wielowarstwowych aplikacji opartych o bazy danych
- Programowanie wyzwalaczy, procedur zapisanych, użycie perspektyw
- Zastosowania transakcji
- Teorię baz danych: reguły Codda, normalizację
- Język SQL

Dr Mark Whitehorn dysponuje ogromną wiedzą w dziedzinie teorii relacyjnych baz danych. Dzięki cyklowi artykułów w brytyjskim magazynie Personal Computer World udało mu się przybliżyć ją tysiącom użytkowników. *„Po prostu doskonała. Wyjaśniając tajniki zagadnień związanych z relacyjnymi bazami danych, Mark Whitehorn oraz Bill Marklyn osiągnęli o wiele więcej niż inni autorzy. Uczynili ten temat ciekawym, a nawet wręcz zabawnym, co stawia ich poza zasięgiem jakiejkolwiek konkurencji".*
— Neil Fawcett, Edytor techniczny, VNU Business Publications

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

Spis treści

Przedmowa do drugiego wydania 9

Rozdział 1. Wstęp 11
Czym jest baza danych? 11
Bazy danych a systemy zarządzania nimi 12
Systemy obsługi relacyjnych baz danych 12
Cel powstania książki 13
Kto powinien przeczytać tę książkę? 15
Struktura książki 16
Dodatkowe uwagi 17
Podziękowania 18

Część I Prosta baza danych 19

Rozdział 2. Wstęp do części pierwszej 21
Tabele 21
Formularze 22
Zapytania 22
Raporty 23

Rozdział 3. Tabele 25
Wiersze i kolumny — rekordy i pola 26
Tworzenie tabeli 28
Typy danych 29
Rozmiar pola 34
Ogólne porady dotyczące projektowania tabel 35
Tabele podstawowe 39

Rozdział 4. Formularze 41
Wykorzystanie kilku formularzy obsługujących pojedynczą tabelę 44
Pola tekstowe udostępniane tylko do odczytu 44
Pola tekstowe zawierające połączone dane z kilku pól 45
Nie wszystkie pola tabeli muszą występować w formularzu 46
Kontrola wprowadzanych danych 47
Wykorzystanie formularzy może być kontrolowane 47
Formularze mogą być stronami WWW 47
Podsumowanie 48

Rozdział 5. Zapytania 51
Pobieranie danych za pomocą zapytań 51
Zapytania, tabele wynikowe oraz tabele podstawowe 52

Wyliczanie danych ... 56
Inne rodzaje zapytań ... 57
Graficzne narzędzia służące do tworzenia zapytań ... 57
SQL i perspektywy ... 58

Rozdział 6. Raporty ... 59

Rozdział 7. Podsumowanie części pierwszej ... 61

Część II Jednoużytkownikowa baza danych, zbudowana z wielu tabel ... 65

Rozdział 8. Wstęp do części drugiej ... 67

Rozdział 9. Problemy z pojedynczymi tabelami ... 69
Nadmiarowość danych ... 70
Błędy typograficzne ... 70
Aktualizacja danych ... 71
Modyfikacja danych ... 72
Podsumowanie ... 72

Rozdział 10. Zastosowanie kilku tabel ... 75
Nadmiarowość danych ... 77
Błędy typograficzne ... 79
Aktualizacja danych ... 79
Modyfikacja danych ... 80

Rozdział 11. Współdziałanie wielu tabel ... 81
Bazy danych są zaprojektowane w celu modelowania świata rzeczywistego ... 82

Rozdział 12. Prawidłowy projekt tabel ... 83
Kilka słów na temat normalizacji i modelowania związków encji ... 85
Identyfikacja klas obiektów ... 85

Rozdział 13. Związki w świecie rzeczywistym ... 89
Związki typu jeden do wielu ... 89
Związki typu jeden do jednego ... 90
Związki typu wiele do wielu ... 90
Brak związku ... 90

Rozdział 14. Modelowanie związków ... 91
Klucze główne ... 93
Wykorzystanie kilku kolumn w charakterze klucza głównego ... 95
Wybór właściwego klucza głównego ... 96
Definiowanie klucza głównego ... 97
Klucze obce ... 97
Definiowanie klucza obcego ... 98
Częściowe podsumowanie ... 98
Klucz główny ... 99
Klucz obcy ... 99
Złączenia ... 99
Związki typu „jeden do wielu” ... 99
Związki typu „jeden do jednego” ... 102
Związki typu „wiele do wielu” ... 104
Ogólne informacje na temat złączeń ... 111

Rozdział 15. Ponowna analiza czterech elementów baz danych 117
Tabele 120
Zapytania 121
Formularze 127
Raporty 128

Rozdział 16. Integralność danych 131
Integrowanie danych — opłacalny wysiłek 131
Błędy integralności danych i ich przyczyny 132
Błędy w unikalnych danych w ramach pojedynczego rekordu 133
Błędy w standardowych danych w ramach pojedynczego rekordu 133
Błędy pomiędzy danymi w różnych polach 135
Błędy pomiędzy kluczami w różnych tabelach 136
Inne zagadnienia dotyczące integralności 142
Definiowanie zasad integralności danych w systemie 143
Deklarowana i proceduralna integralność odwołań 144

Rozdział 17. Budowanie aplikacji wykorzystującej bazę danych 147
Wykorzystanie narzędzi graficznych, makr oraz języków skryptowych 149
Tworzenie prostego interfejsu 149
Wykorzystanie narzędzi graficznych 150
Wykorzystanie makra 151
Wykorzystanie języka programowania 152
Które z rozwiązań jest najlepsze? 154
Inne języki — SQL 156

Rozdział 18. Podsumowanie części drugiej 157

Część III Bazy danych w środowisku wieloużytkownikowym 159

Rozdział 19. Architektura baz danych 161
Siedem elementów architektury 161
Element pierwszy 162
Element drugi 162
Element trzeci 162
Element czwarty 162
Element piąty 163
Element szósty 163
Element siódmy 163
Interfejs aplikacji na komputerze klienta, baza danych na serwerze 164
Architektura klient-serwer (dwuwarstwowa) 167
Architektura trójwarstwowa (wielowarstwowa) 169
Aplikacje internetowe 170
Wybór odpowiedniej architektury 172
Podsumowanie 173

Rozdział 20. Bardziej skomplikowane projekty baz danych 175
Model użytkownika 177
Model logiczny 177
Model fizyczny 178
Model logiczny i fizyczny w praktyce 179
Podsumowanie częściowe 182
Kolejna wielka zaleta narzędzi CASE 183
Różnice pomiędzy modelem logicznym a fizycznym 184
Normalizacja 186

Inżynieria odwrotna ... 187
Różne metodologie ... 187

Rozdział 21. Wyzwalacze, zapisane procedury oraz perspektywy ... 189
Wyzwalacze ... 189
Terminologia wyzwalaczy ... 190
Typowe zastosowanie wyzwalaczy, wywoływanych przed i po zdarzeniu ... 190
Więcej informacji na temat wyzwalaczy ... 191
Zapisane procedury ... 191
Wyzwalacze i zapisane procedury — podsumowanie ... 192
Perspektywy ... 193

Rozdział 22. Transakcje, dzienniki, kopie zapasowe, blokowanie i współbieżność ... 197
Transakcje ... 197
Wycofanie ... 198
Dzienniki ... 198
Przywrócenie transakcji ... 201
Archiwizowanie dzienników ... 201
Lokalizacja ... 202
Strategie wykonywania kopii zapasowych ... 202
Zalecenia ... 203
Inne możliwości ... 204
Blokowanie ... 204
Zakleszczenia ... 205
Współbieżność ... 206
Blokowanie wierszy i stron ... 207
Co czeka nas w kolejnych rozdziałach ... 207
Rozwiązanie testu ... 207

Część IV Tematy związane z bazami danych ... 209

Rozdział 23. Relacyjne i nierelacyjne bazy danych ... 211
Wiele tabel a relacyjność baz danych ... 211
Nazewnictwo ... 212

Rozdział 24. Reguły Codda ... 215
Do czego potrzebna jest znajomość reguł Codda ... 215
Zwięzłość a czytelność ... 216
Krótkie wprowadzenie ... 216
Reguły ... 216
Podsumowanie ... 226

Rozdział 25. Normalizacja ... 229
Normalizacja ... 229
Zależność funkcyjna ... 231
Definicja wymagań ... 232
Pierwsza forma normalna ... 234
Druga forma normalna ... 236
Odpowiedzi ... 239
Trzecia forma normalna ... 240
Podsumowanie częściowe ... 242
Pierwsza forma normalna (1NF lub pierwszy poziom normalizacji) ... 242
Druga forma normalna (2NF lub drugi poziom normalizacji) ... 242
Trzecia forma normalna (3NF lub trzeci poziom normalizacji) ... 242

Dalsze postępowanie ... 242
Przykład praktyczny ... 243
Normalizacja a usuwanie wszelkiej nadmiarowości ... 245
Podsumowanie ... 251

Rozdział 26. Katalog systemowy ... 253
Katalog systemowy ... 253

Rozdział 27. Operacje na danych ... 255
Operatory relacyjne ... 255
Selekcja ... 257
Projekcja ... 257
Suma ... 258
Różnica ... 259
Przecięcie ... 260
Iloczyn ... 260
Złączenie ... 262
Podział ... 262
Podsumowanie ... 264

Rozdział 28. SQL ... 267
SELECT oraz FROM ... 270
DISTINCT ... 270
WHERE ... 271
Warunki ... 272
ORDER BY ... 275
Wzorce dopasowań ... 277
Podzapytania ... 278
Funkcje wbudowane ... 279
GROUP BY ... 282
GROUP BY...HAVING ... 287
Praca z wieloma tabelami ... 289
Złączenia wewnętrzne ... 293
Złączenia zewnętrzne ... 293
Suma tabel ... 295
Podsumowanie instrukcji SELECT ... 298
INSERT ... 299
UPDATE ... 301
DELETE ... 303
Analiza przypadku ... 304
DISTINCTROW ... 306
Podsumowanie ... 310

Rozdział 29. Dziedziny 311
Rozdział 30. Indeksy — przyśpieszanie działania bazy danych 313
Rozdział 31. Znaczenie wartości NULL 319
Rozdział 32. Klucze główne 323

Dodatki 327

Dodatek A Słowniczek 329
Skorowidz 331

Rozdział 14.

Modelowanie związków

Udało nam się umieścić klasy obiektów w tabelach bazy danych. Znamy również sieć zależności, istniejących pomiędzy obiektami. Kolejnym zadaniem będzie odwzorowanie tych związków w bazie.

Do realizacji tego zadania będą nam potrzebne pewne narzędzia:

- klucze — główne oraz obce;
- złączenia.

Te dwa pojęcia określają dwa różne typy narzędzi, udostępnianych przez relacyjne bazy danych, lecz najczęściej oba są wykorzystywane jednocześnie. Na przykład: przed zdefiniowaniem złączenia należy utworzyć jeden lub kilka kluczy głównych, zaś dopiero zbudowanie złączenia nadaje sens istnieniu klucza głównego. Nie należy przejmować się tym, że nazwy tych mechanizmów brzmią dla nas niezrozumiale; w rzeczywistości są to dość proste pojęcia. Zrozumienie ich jest jednak bardzo ważnym krokiem, bowiem dzięki nim wiele zagadnień relacyjnych baz danych nabiera głębszego sensu. Trudno byłoby wyobrazić sobie istnienie relacyjnych baz danych bez kluczy i złączeń.

Zagadnienia dotyczące kluczy i złączeń najprościej omawiać na podstawie przykładu. Jednym z najczęściej stosowanych związków są związki typu „jeden do wielu", omówione w poprzednim rozdziale. Przedstawiliśmy tam związek pomiędzy klientami oraz zamówieniami. Wykorzystamy ten przykład także i tutaj.



Pamiętajmy o stosowanej konwencji nazewnictwa: `WIELKIE_LITERY` określają nazwy tabel, natomiast `MieszaneLitery` określają nazwy kolumn. Odwołania do kolumn w tabeli są zapisywane jako nazwa tabeli, po której widnieje kropka, a następnie nazwa kolumny. Dlatego zapis `KLIENCI.NrKlienta` określa kolumnę `NrKlienta` tabeli `KLIENCI`.

Jedna z zaprezentowanych tu tabel przedstawia uproszczoną wersję tabeli `ZAMOWIENIA`, zawierającą odnośniki do tabeli `KLIENCI` oraz `PRACOWNICY`:

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
3	Henryk	Bezbożny
4	Janina	Kozłowska

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Dostawca	Cena	Towar
1	*1*	*2*	Jan Strug	235,00 zł	Biurko
2	*4*	*1*	Fab. Mebli	234,00 zł	Krzesło
3	*1*	*3*	Jan Strug	415,00 zł	Stół
4	*2*	*4*	Błysk s.c.	350,00 zł	Lampa
5	*3*	*2*	Fab. Mebli	234,00 zł	Krzesło
6	*2*	*4*	Błysk s.c.	350,00 zł	Lampa
7	*2*	*2*	Jan Strug	235,00 zł	Biurko

PRACOWNICY

NrPracownika	Imie	Nazwisko	DataUrodzenia	DataZatrudnienia
1	Jan	Kowalski	12 kwi 1966	01 maj 1999
2	Maria	Nowak	21 mar 1977	01 sty 2000
3	Zygmunt	Zdun	01 maj 1977	01 kwi 2002
4	Zofia	Kwiatkowska	03 kwi 1966	01 kwi 2002
5	Stefan	Czarnecki	12 kwi 1966	01 kwi 2002
6	Władysław	Mały	01 maj 1967	01 maj 2004

Tabele te odnaleźć można w pliku *R14.MDB*. W celu zwiększenia ich czytelności klucze główne zostały zapisane **wytłuszczoną czcionką**, natomiast klucze obce — *kursywą*.



Zdaję sobie sprawę z tego, że tabela ZAMOWIENIA jest daleka od doskonałości, ponieważ nadal zawiera informacje nadmiarowe (dotyczące towarów). Zamiast tego należałoby zastosować osobną tabelę TOWARY. Jest to jednak zabieg celowy — chciałem uniknąć nadmiernej komplikacji. Udoskonaleniem tabeli ZAMOWIENIA zajmiemy się w dalszej części niniejszego rozdziału.

Po bliższym zapoznaniu się z tabelami KLIENCI oraz ZAMOWIENIA możemy zauważyć, że dane łączące te tabele, znajdują się w dwóch kolumnach (po jednej w każdej z tabel) o tej samej nazwie (NrKlienta). Bez trudu można domyślić się, że liczba 2 w kolumnie NrKlienta, w pierwszym wierszu tabeli ZAMOWIENIA, oznacza, że pani Alicja Kwiatkowska zakupiła biurko. Liczba 2 w tym miejscu jest wykorzystana w charakterze wskaźnika rekordu w tabeli KLIENCI, zawierającego dane pani Alicji. Analogicznie, na podstawie innych danych możemy dowiedzieć się, że zamówienie zrealizował pracownik Jan Kowalski.

W celu wyjaśnienia mechanizmu modelowania związków możemy posłużyć się zarówno przykładowymi związkami pomiędzy tabelą ZAMOWIENIA a tabelą KLIENCI, jak i związkami pomiędzy tabelą ZAMOWIENIA a tabelą PRACOWNICY, ponieważ w obydwu przypadkach związki te są tworzone i obsługiwane na identycznych zasadach. Na razie zajmiemy się związkami pomiędzy tabelą ZAMOWIENIA a tabelą KLIENCI, lecz w odpowiednim czasie wrócimy jeszcze do tabeli PRACOWNICY.

Aby wykorzystać kolumny NrKlienta w charakterze mechanizmu definiującego związki pomiędzy dwiema tabelami, każda z kolumn w o nazwie NrKlienta musi spełniać określone założenia. W skrócie można powiedzieć, że kolumna NrKlienta w tabeli KLIENCI musi być **kluczem głównym**, natomiast kolumna NrKlienta w tabeli ZAMOWIENIA musi być **kluczem obcym**.

W celu utworzenia związku „jeden do wielu" pomiędzy dwiema tabelami należy utworzyć w jednej z nich klucz główny, w drugiej natomiast klucz obcy. Klucz główny określa stronę związku określaną przez „jeden", natomiast klucz obcy definiuje stronę związku określaną przez „wiele". Wartości kluczy są przez nas wykorzystywane do modelowania związków pomiędzy klientami i zamówieniami w świecie rzeczywistym.

Pojęcia kluczy głównych i obcych są bardzo istotne. Poświęćmy im więc nieco uwagi.

Klucze główne

Wymagania dotyczące klucza głównego są dosyć jasne. Moglibyśmy wymienić je teraz po kolei, lecz wydaje się, że wydedukowanie ich przyniesie nieco więcej satysfakcji, a przede wszystkim będzie miało lepszy skutek pedagogiczny. Wiele zagadnień w relacyjnej teorii baz danych można dość łatwo wywieść za pomocą logicznego rozumowania.

Pamiętamy o tym, że kolumna NrKlienta w tabeli KLIENCI zawiera klucze główne tej tabeli. Zawartość pól w tej kolumnie identyfikuje poszczególnych klientów. Klucz o wartości 1 identyfikuje Edwarda Dzikiego, klucz o wartości 2 Alicję Kwiatkowską, itd. Co stałoby się, gdyby klucz, identyfikujący Alicję Kwiatkowską, został zmieniony na wartość 1? Tabele KLIENCI oraz ZAMOWIENIA (patrz następna strona) wyglądałyby w tej sytuacji następująco:

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
1	Alicja	Kwiatkowska
3	Henryk	Bezbożny
4	Janina	Kozłowska

Poprzez zamianę wartości jednego pola w tabeli KLIENCI nie można określić tego, komu należy przesłać rachunek za zamówienia numer 2 — nie wiemy, czy krzesło kupiła Alicja, czy Edward. (Dodatkowy problem występuje z zamówieniami o numerach 1, 5

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Dostawca	Cena	Towar
1	1	2	Jan Strug	235,00 zł	Biurko
2	4	1	Fab. Mebli	234,00 zł	Krzesło
3	1	3	Jan Strug	415,00 zł	Stół
4	2	4	Błysk s.c.	350,00 zł	Lampa
5	3	2	Fab. Mebli	234,00 zł	Krzesło
6	2	4	Błysk s.c.	350,00 zł	Lampa
7	2	2	Jan Strug	235,00 zł	Biurko

oraz 7, lecz to zagadnienie omówimy w podrozdziale, w którym zajmować się będziemy kluczami obcymi.) Pierwszą zasadą dotyczącą kluczy głównych, którą udało nam się wydedukować, jest więc konieczność zachowania **unikalności wszystkich kluczy głównych** w tabeli, bowiem jakiekolwiek duplikaty nie będą tolerowane.

Kolejną sytuacją, którą warto rozpatrzyć, jest brak wartości klucza głównego w tabeli.

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
	Henryk	Bezbożny
4	Janina	Kozłowska

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Dostawca	Cena	Towar
1	1	2	Jan Strug	235,00 zł	Biurko
2	4	1	Fab. Mebli	234,00 zł	Krzesło
3	1	3	Jan Strug	415,00 zł	Stół
4	2	4	Błysk s.c.	350,00 zł	Lampa
5	3	2	Fab. Mebli	234,00 zł	Krzesło
6	2	4	Błysk s.c.	350,00 zł	Lampa
7	2	2	Jan Strug	235,00 zł	Biurko

W kolumnie KLIENCI.NrKlienta brakuje wartości dla klienta o nazwisku Henryk Bezbożny. Czy w takim razie nie będzie on musiał płacić za zamówienia nr 3? Niewiele lepsza będzie sytuacja, gdy tabele będą wyglądać tak:

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
	Henryk	Bezbożny
4	Janina	Kozłowska

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Dostawca	Cena	Towar
1	1	2	Jan Strug	235,00 zł	Biurko
2	4	1	Fab. Mebli	234,00 zł	Krzesło
3	1		Jan Strug	415,00 zł	Stół
4	2	4	Błysk s.c.	350,00 zł	Lampa
5	3	2	Fab. Mebli	234,00 zł	Krzesło
6	2	4	Błysk s.c.	350,00 zł	Lampa
7	2	5	Jan Strug	235,00 zł	Biurko

W tym przypadku odpowiedź nadal nie jest oczywista. Najlepszym sposobem na uniknięcie takich niejasności jest zadbanie o to, by wszystkie pola w kluczu głównym posiadały określoną wartość. Brak wartości w bazie danych określa się **wartością NULL** (zwaną również **wartością pustą** lub **wartością zerową**). Jest to dość specyficzna wartość, a jej obsługa w bazach danych wiąże się z dość ciekawymi kwestiami. Rozdział 31., umieszczony w części IV, jest poświęcony w całości właśnie zagadnieniom związanym z wartością `NULL`.

W ten sposób udało nam się określić drugą zasadę dotyczącą kluczy głównych. Klucze główne nie mogą się powtarzać oraz nie mogą zawierać pustych wartości.

Systemy zarządzania bazami danych, takie jak MS Access, obsługują klucze główne z uwzględnieniem tych zasad. Należy tylko wskazać systemowi, które z kolumn zawierają klucze główne, a zastosuje on zabezpieczenia przed złamaniem tych reguł. Można się zastanawiać, czy rzeczywiście system jest w stanie sprawdzić, czy wartość pola będącego kluczem głównym, nie występuje już w tabeli, szczególnie w przypadku tabel zawierających setki tysięcy wierszy danych? W rzeczywistości mechanizmy obsługi kluczy głównych w porządnych relacyjnych systemach zarządzania bazą danych są w stanie dokonać takiego sprawdzenia w sposób tak szybki, że nie będzie to miało widocznego wpływu na czas, jaki zajmuje dopisanie wiersza do tabeli.

Wykorzystanie kilku kolumn w charakterze klucza głównego

Przedstawione jak dotąd w tym podrozdziale wymagania (unikalność oraz konieczność określenia wartości) dotyczące kluczy głównych, nie ograniczają ich konstrukcji tylko do jednej kolumny. Możemy na przykład wykorzystać kolumny `Imie` oraz `Nazwisko`. Naruszenie zasady unikalności danych nastąpi wyłącznie w sytuacji, gdy pola obydwu kolumn będą identyczne w rozpatrywanym rekordzie danych. Możemy więc posiadać w naszej tabeli osoby o nazwisku Jan Kowalski oraz Jan Kowal, lecz nie możemy zapisać dwóch osób o nazwisku Jan Kowalski. Wykorzystanie więcej niż jednej kolumny w charakterze klucza głównego jest więc jak najbardziej dopuszczalne, lecz z reguły nie jest zalecane. Bywają jednak przypadki, w których takie rozwiązanie ma pierwszorzędne znaczenie. Jedno z takich zastosowań omówimy w podrozdziale, w którym zajmiemy się związkami typu „wiele do wielu”.

Wybór właściwego klucza głównego

Wybór klucza głównego dla tabeli może być trudny, dodatkowo sprawę tę komplikuje możliwość zastosowania do tego kilku kolumn tabeli.

Jak już wspomnieliśmy, istnieją sytuacje, w których wykorzystanie kilku kolumn w charakterze klucza głównego stanowi jedyne rozsądne rozwiązanie. Jeśli jednak nie można znaleźć bezpośredniego powodu, dla którego mielibyśmy zdecydować się na klucz główny złożony z kilku kolumn, należy wybrać rozwiązanie na bazie pojedynczej kolumny. Nie jest to zasada niepodważalna — to tylko dobra rada. Klucze główne zbudowane na podstawie pojedynczej kolumny są łatwiejsze w obsłudze i z reguły działają szybciej. Oznacza to, że w przypadku zastosowania klucza głównego zbudowanego na podstawie pojedynczej kolumny, zapytania w bazie danych będą wykonywane szybciej.

Kolejny problem stanowi właściwy wybór kolumny. Podstawowym kryterium wyboru jest tutaj zapewnienie unikalności wartości w kolumnie. W przypadku tabeli pracowników wybór kolumny `Imie` jest oczywiście dość kiepski, ponieważ istnieją wielkie szanse na to, że ich imiona będą się powtarzać.



Jedna z anegdot na temat Billa Gatesa, założyciela i wieloletniego prezesa firmy Microsoft głosi, że jest to człowiek o skłonnościach paranoidalnych. Jednym z objawów choroby miałby być zakaz zatrudniania w firmie osób o takim samym imieniu, jakie nosi Prezes. W jednej z wersji tej anegdoty utrzymuje się, że zanim zatrudniono kolejnego Billa, liczba pracowników Microsoftu wynosiła powyżej pięciuset osób. Historia ta jest bardzo interesująca, lecz, niestety, nieprawdziwa. Bill Marklyn, współautor tej książki, został zatrudniony w firmie Microsoft na długo przed osiągnięciem przez nią wspomnianej liczby pracowników. Można z tego wysnuć wniosek, że w Microsofcie nie stosuje się klucza głównego w tabeli `PRACOWNICY`, zbudowanego na podstawie kolumny `Imie`.

Najprostszym i powszechnie stosowanym sposobem zapewnienia unikalności klucza głównego jest wykorzystanie do tego celu samego systemu bazy danych. Większość relacyjnych systemów zarządzania bazą danych posiada mechanizmy umożliwiające automatyczne generowanie unikalnych identyfikatorów dla każdego dopisywanego wiersza. Access na przykład udostępnia w tym celu specjalny typ danych o nazwie `Autonumerowanie`. Jest to doskonałe rozwiązanie w przypadku identyfikacji obiektów, takich jak zamówienia, pracownicy, itd. Warto jednak zorientować się, czy w tabeli nie jest już zapisana informacja, zapewniająca unikalność danych dzięki swojej specyficznej charakterystyce. Taką informacją w przypadku pracowników może być na przykład numer PESEL. Jest to z założenia unikalny i niezmienny identyfikator każdego obywatela Rzeczpospolitej Polskiej, więc jest idealny w celu zdefiniowania klucza głównego tabeli (w każdym razie w firmie zatrudniającej wyłącznie obywateli naszego kraju)[1].

[1] Należy oczywiście wziąć pod uwagę realia. Jeśli zatrudnienie w firmie obcokrajowców wchodzi w rachubę, zastosowanie numeru PESEL nie jest uzasadnione — *przyp. tłum.*



Generowanie naprawdę unikalnych identyfikatorów nie jest zadaniem tak prostym, jakim się może wydawać. Więcej informacji na ten temat można znaleźć w rozdziale 32.

Definiowanie klucza głównego

Definiowanie klucza głównego tabeli jest bardzo ważnym elementem w procesie jej konstruowania. Pomimo, że nie jest to książka na temat programu MS Access, sposób zdefiniowania klucza głównego zademonstrujemy na przykładzie tego programu. Ważne jest również spostrzeżenie różnicy pomiędzy kluczami głównymi a obcymi. Klucz główny należy zdefiniować na etapie definiowania struktury tabeli. Tworzenie klucza obcego nie musi natomiast być dokonywane w sposób jawny. Wykorzystanie wartości określonej kolumny w charakterze klucza obcego może zostać dokonane dopiero podczas tworzenia złączenia tabel.

W jaki sposób możemy zatem zdefiniować klucz główny w programie MS Access? Na etapie tworzenia struktury tabeli zaznaczamy odpowiedni wiersz. Następnie klikamy przycisk z ikoną klucza na pasku narzędzi. Najczęściej kolumna definiująca klucz główny jest umieszczana jako pierwsza w tabeli, nie jest to jednak konieczne. Jeśli wystąpi potrzeba zdefiniowania klucza głównego na większej ilości kolumn, zaznaczamy odpowiednie kolumny i klikamy przycisk z ikoną klucza. Warto zwrócić uwagę na to, że w nomenklaturze zastosowanej w polskiej wersji programu MS Access, klucze główne noszą nazwę kluczy podstawowych.

Klucze obce

Pozostaniemy przy naszym przykładzie, wykorzystującym tabele `KLIENCI` oraz `ZAMOWIENIA`. Przejdźmy teraz do omówienia kluczy obcych w modelowaniu związków typu „jeden do wielu". Klucz obcy jest po prostu referencją pewnego klucza głównego. W naszym przypadku kluczem obcym jest kolumna `ZAMOWIENIA.NrKlienta`.

Ponownie zastosujemy regułę intuicyjnego wyodrębniania zasad regulujących istnienie kluczy obcych. Weźmy pod uwagę wartości z poniższych tabel.

`KLIENCI`

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
3	Henryk	Bezbożny
4	Janina	Kozłowska

Zwróćmy uwagę na ostatni wiersz tabeli `ZAMOWIENIA`. Zawiera on wartość `5` w kolumnie `NrKlienta`. Jest to nieco dziwne, ponieważ w tabeli `KLIENCI` nie istnieje wiersz o takiej wartości w kolumnie `NrKlienta`. W tym przypadku nie jesteśmy w stanie określić

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Dostawca	Cena	Towar
1	1	2	Jan Strug	235,00 zł	Biurko
2	4	1	Fab. Mebli	234,00 zł	Krzesło
3	1	3	Jan Strug	415,00 zł	Stół
4	2	4	Błysk s.c.	350,00 zł	Lampa
5	3	2	Fab. Mebli	234,00 zł	Krzesło
6	2	4	Błysk s.c.	350,00 zł	Lampa
7	2	5	Jan Strug	235,00 zł	Biurko

adresata rachunku za zamówienie numer 7. Można się zatem domyślić, że tego typu sytuacja jest nie do przyjęcia w bazie danych. Stąd właśnie wynika pierwsza zasada, dotycząca kluczy obcych. Zasada ta wymusza stosowanie w kluczach obcych wyłącznie takich wartości, jakie występują w odpowiednim dla nich kluczu głównym.

Definiowanie klucza obcego

Klucze obce są definiowane w ramach procesu definicji złączenia tabel. Proces ten zostanie omówiony w dalszej części niniejszego rozdziału.

Częściowe podsumowanie

Zanim przejdziemy do kolejnych zagadnień, podsumujmy informacje, z którymi zapoznaliśmy się do tej pory. Ponownie prezentujemy znane nam już tabele:

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
3	Henryk	Bezbożny
4	Janina	Kozłowska

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Dostawca	Cena	Towar
1	1	2	Jan Strug	235,00 zł	Biurko
2	4	1	Fab. Mebli	234,00 zł	Krzesło
3	1	3	Jan Strug	415,00 zł	Stół
4	2	4	Błysk s.c.	350,00 zł	Lampa
5	3	2	Fab. Mebli	234,00 zł	Krzesło
6	2	4	Błysk s.c.	350,00 zł	Lampa
7	2	2	Jan Strug	235,00 zł	Biurko

Klucz główny

Każda tabela w relacyjnej bazie danych musi posiadać klucz główny. Klucz taki składa się z co najmniej jednego pola (kolumny). Żadna z wartości klucza głównego nie może być pusta. Każdy z wierszy w tabeli musi posiadać unikalną wartość klucza głównego. Klucz główny jest definiowany na etapie definicji tabeli.

Klucz obcy

Klucze obce nie są wymaganym elementem tabel. Innymi słowy, każda tabela musi posiadać zdefiniowany klucz główny, lecz nie wszystkie muszą definiować klucze obce. Jeśli istnieje związek pomiędzy dwiema tabelami, jedna z nich musi posiadać klucz obcy, na podstawie którego pobierane są odpowiednie dane z drugiej tabeli. W praktyce bywa tak, że większość tabel posiada klucze obce i zupełnie dopuszczalne jest występowanie więcej niż jednego klucza obcego w tabeli. Jeśli tak jest w istocie, tabela musi posiadać związki z kilkoma innymi tabelami. Definicja klucza obcego następuje podczas definicji złączenia.

Złączenia

Jak wspomnieliśmy na początku niniejszego rozdziału, narzędziami, służącymi do określania związków w ramach baz danych, są:

- klucze (główne i obce),
- złączenia.

Nadszedł czas na omówienie drugiego z wymienionych narzędzi. W poprzednim rozdziale zostały wymienione trzy możliwe typy związków, jakie można modelować w bazie danych:

- „jeden do wielu”,
- „wiele do wielu”,
- „jeden do jednego”.

Nasze rozważania dotyczące złączeń rozpoczniemy od najczęściej spotykanych związków typu „jeden do wielu”.

Związki typu „jeden do wielu”

Załóżmy, że posiadamy dwie tabele:

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
3	Henryk	Bezbożny
4	Janina	Kozłowska

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Dostawca	Cena	Towar
1	*1*	*2*	Jan Strug	235,00 zł	Biurko
2	*4*	*1*	Fab. Mebli	234,00 zł	Krzesło
3	*1*	*3*	Jan Strug	415,00 zł	Stół
4	*2*	*4*	Błysk s.c.	350,00 zł	Lampa
5	*3*	*2*	Fab. Mebli	234,00 zł	Krzesło
6	*2*	*4*	Błysk s.c.	350,00 zł	Lampa
7	*2*	*2*	Jan Strug	235,00 zł	Biurko

Czy można określić typ związku pomiędzy tymi tabelami na podstawie ich zawartości? Odpowiedź brzmi: tak i nie. Nie możemy być absolutnie pewni co do typu związku, lecz możemy dokonać pewnych, bardzo prawdopodobnych założeń.

Możemy założyć, że `ZAMOWIENIA.NrZamowienia` jest kluczem głównym. Pierwszą wskazówką, pozwalającą nam na takie założenie, jest nazwa kolumny; drugą stanowią unikalne wartości w ramach tej kolumny. Podobne założenie możemy przyjąć na temat kolumny `KLIENCI.NrKlienta`.

Można zaobserwować, że wartości w kolumnie `ZAMOWIENIA.NrKlienta` odpowiadają wartościom w kolumnie `KLIENCI.NrKlienta`, dlatego bardzo prawdopodobne jest, że kolumna `ZAMOWIENIA.NrKlienta` jest kluczem obcym.

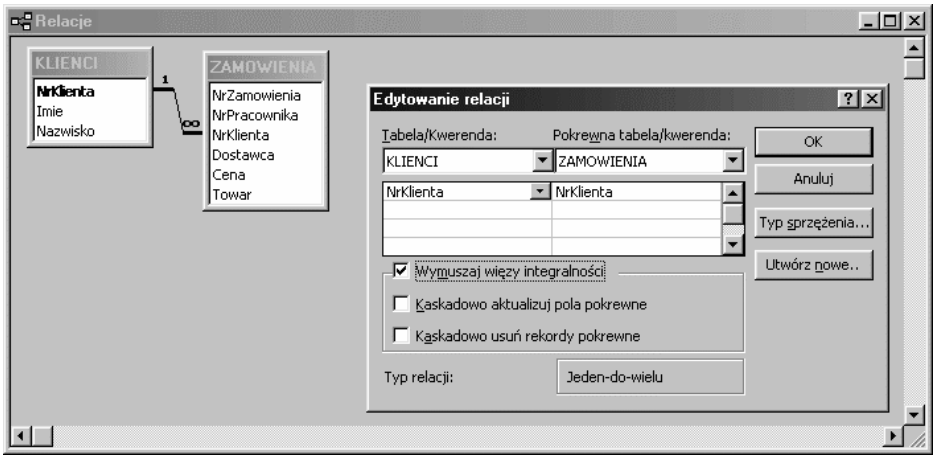
Czemu mają służyć powyższe dociekania? Nie namawiam czytelnika do tego, aby oglądał zawartość tabel w celu odgadnięcia typów związków pomiędzy nimi. Celem powyższego ćwiczenia jest uświadomienie czytelnikowi szczególnych własności, które pomagają w procesie definiowania związków pomiędzy tabelami i zarządzania nimi.

Możemy więc zająć się procesem modelowania złączenia typu „jeden do wielu”. Proces ten składa się z następujących kroków:

- Podejmujemy decyzję umieszczenia w bazie danych dwóch tabel, reprezentujących klasy obiektów zamówień oraz klientów.
- Związek pomiędzy tymi klasami obiektów jest typu „jeden do wielu”, co oznacza, że klient może mieć związek z wieloma zamówieniami.
- Tworzymy tabele, reprezentujące wspomniane klasy obiektów, nadając im nazwy `KLIENCI` oraz `ZAMOWIENIA`.
- Każda z wymienionych tabel będzie miała klucz główny, przede wszystkim dlatego, że taki jest wymóg w stosunku do tabel w relacyjnych bazach danych. Klucze główne nazwiemy `KLIENCI.NrKlienta` oraz `ZAMOWIENIA.NrZamowienia`.

- Ponieważ nasze tabele mają za zadanie odzwierciedlać obiekty ze świata rzeczywistego, mamy zamiar modelować również związki pomiędzy reprezentowanymi klasami obiektów. W naszym przypadku będzie to związek typu „jeden do wielu".
- W celu umożliwienia realizacji związku w bazie danych musimy utworzyć odpowiedni klucz obcy w tabeli reprezentującej stronę określaną przez „wiele". W naszym przypadku będzie to tabela `ZAMOWIENIA`.
- Kolumna klucza obcego może mieć dowolną nazwę, jednak powszechną praktyką jest nadawanie kolumnom tego typu takich samych nazw, jakie posiadają kolumny klucza głównego, do którego się odnoszą.
- Do tabeli `ZAMOWIENIA` dodajemy nową kolumnę o nazwie `NrKlienta`.
- Wskazujemy systemowi bazy danych istnienie związku typu „jeden do wielu" pomiędzy rozpatrywanymi tabelami.

W przypadku aplikacji MS Access zdefiniowanie złączenia pomiędzy tabelami polega na otwarciu *Edytora relacji* (ponieważ związki w polskiej wersji MS Access noszą nazwę *relacji*, co jest w pewnym konflikcie z przyjętą terminologią dotyczącą relacyjnych baz danych), dodaniu do relacji dwóch tabel, a następnie przeciągnięciu nazwy kolumny `KLIENCI.NrKlienta` na `ZAMOWIENIA.NrKlienta`. Po otworzeniu się okna zatytułowanego *Edytowanie relacji* należy zaznaczyć opcję *Wymuszaj więzy integralności*. Utworzenie związku zatwierdzamy przez kliknięcie przycisku *OK*. Osoby zainteresowane znaczeniem wspomnianych więzów integralności odsyłam do rozdziału 16., zatytułowanego „Integralność danych".



Rysunek 14.1.

W operacji przedstawionej na rysunku 14.1 zdefiniowano złączenie tabel w bazie danych. Efektem dodatkowym było nadanie kolumnie `ZAMOWIENIA.NrKlienta` roli klucza obcego. W konsekwencji tej zmiany system odmówi wprowadzenia do kolumny `ZAMOWIENIA.NrKlienta` jakiejkolwiek wartości nie występującej w kluczu głównym, do którego odnosi się klucz obcy, reprezentowany przez tę kolumnę.

Przedstawiony przez nas proces definiowania złączenia w systemie Access unaocznił, w jaki sposób złączenia oraz klucze są wzajemnie ze sobą powiązane. W celu zdefiniowania złączenia, podobnego do omówionego powyżej, należy wcześniej zdefiniować klucz główny, a także utworzyć kolumnę, która ma realizować funkcję klucza obcego. Dopiero jednak utworzenie złączenia tabel nada kolumnie rolę klucza obcego.

Należy zdawać sobie sprawę z kilku faktów, dotyczących definiowania złączeń tabel. Złączenie (z wymuszeniem więzów integralności) nie zostanie utworzone w przypadku, gdy:

- kolumna wskazana po stronie „jeden" związku nie jest kluczem głównym,
- typy danych w łączonych kolumnach nie są identyczne,
- w polach klucza obcego (strona „wiele") istnieją wartości, które nie występują w polach klucza głównego (strona „jeden").

Wspomnieliśmy wcześniej o istnieniu możliwości wykorzystania w Accessie kolumn typu `Autonumerowanie` w charakterze kluczy głównych tabel. Kolumny tego typu nie możemy zastosować w charakterze klucza obcego. Typem odpowiadającym typowi `Autonumerowanie`, który może zostać zastosowany w charakterze odpowiedniego klucza obcego, jest typ `Liczbowy`, podtyp `Liczba całkowita długa (Long Integer)`. Nie jest to, jak mogłoby się wydawać, złamanie wymogu identyczności typów danych w kluczach głównych i wskazujących na nie kluczach obcych. Typ `Autonumerowanie` jest specjalnym przypadkiem typu `Liczba całkowita długa (Long Integer)`, z tą różnicą, że wartości w kolumnach tego typu podlegają automatycznemu zwiększaniu podczas dopisywania nowej kolumny.

Związki typu „jeden do jednego"

Związki typu „jeden do jednego" stosowane są raczej dość rzadko. Tworzenie ich jest jednak bardzo proste, ponieważ proces definicji złączeń tabel w takich związkach jest bardzo podobny do definicji złączenia tabel w związkach typu „jeden do wielu". Podobnie jak miało to miejsce w przypadku związków typu „jeden do wielu", należy przeciągnąć nazwę kolumny z klucza głównego na nazwę kolumny klucza obcego. Jedyna różnica polega na tym, że klucz obcy w tym przypadku nie może zawierać zduplikowanych wartości. Istnieją dwie metody zapewnienia unikalności wartości w ramach kolumn kluczy obcych.

Pierwsza metoda polega na utworzeniu klucza obcego tabeli na podstawie jej klucza głównego. Innymi słowy, złączenie nastąpi pomiędzy kluczami głównymi dwóch tabel, z których jeden będzie również pełnił rolę klucza obcego.

Drugi sposób polega na utworzeniu indeksu bez powtórzeń na kolumnie klucza obcego. Taka operacja spowoduje, że jako wartości pól w tej kolumnie nie zostaną przyjęte wartości już występujące. Ten właśnie sposób na zapewnienie unikalności danych w kolumnie omówimy nieco szerzej.

W przykładowej bazie danych każdemu pracownikowi musi zostać przyznany pokój do wyłącznego wykorzystania, lecz ze względu na koszty żaden pracownik nie może mieć do dyspozycji więcej niż jeden pokój.

PRACOWNICY

NrPracownika	Imie	Nazwisko	DataUrodzenia	DataZatrudnienia
1	Jan	Kowalski	12 kwi 1966	01 maj 1999
2	Maria	Nowak	21 mar 1977	01 sty 2000
3	Zygmunt	Zdun	01 maj 1977	01 kwi 2002
4	Zofia	Kwiatkowska	03 kwi 1966	01 kwi 2002
5	Stefan	Czarnecki	12 kwi 1966	01 kwi 2002
6	Władysław	Mały	01 maj 1967	01 maj 2004

POKOJE

NrPokoju	NrPracownika
1	*2*
12	*4*
23	*1*
24	*6*

Kolumna `PRACOWNICY.NrPracownika` jest kluczem głównym, więc w jej przypadku nie są dopuszczalne powtórzenia. Na kolumnie `POKOJE.NrPracownika` w trakcie projektowania tabeli został założony unikalny indeks, więc również w tym przypadku nie są dopuszczalne powtórzenia. Dodatkowo kolumna `POKOJE.NrPracownika` jest kluczem obcym, więc wartości pól tej kolumny muszą odpowiadać istniejącym wartościom pól kolumny odpowiedniego klucza głównego, to znaczy kolumny `PRACOWNICY.NrPracownika`.

Z tego powodu poniższa tabela zawiera nieprawidłową zawartość w kolumnie klucza obcego, ponieważ wartość `9` nie występuje w kolumnie klucza głównego tabeli `PRACOWNICY`:

POKOJE

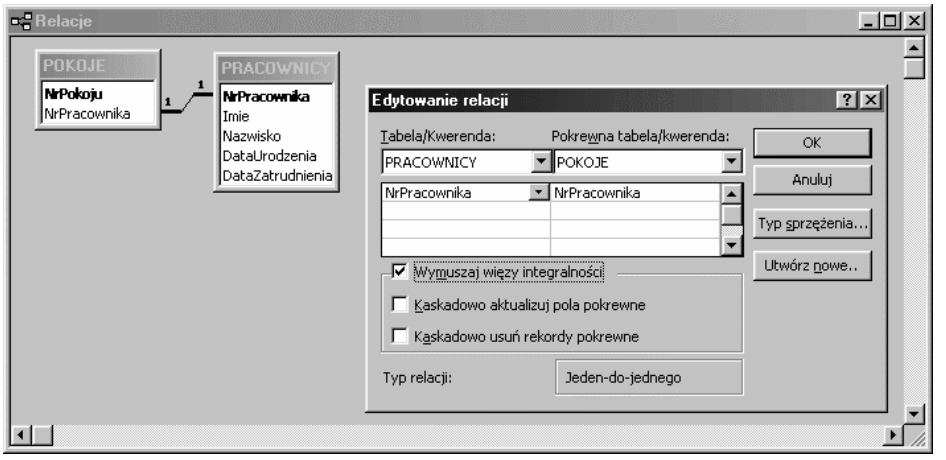
NrPokoju	NrPracownika
1	*2*
12	*4*
23	*1*
24	*9*

Także zawartość tabeli zaprezentowanej poniżej jest niedopuszczalna, ponieważ wartość `2` występuje dwukrotnie w kolumnie `POKOJE.NrPracownika`:

POKOJE

NrPokoju	NrPracownika
1	*2*
12	*4*
23	*1*
24	*2*

Utworzenie związku typu „jeden do jednego” wymaga wykorzystania *Edytora relacji*. Dodajemy do relacji tabele `PRACOWNICY` oraz `POKOJE` (poprzez wykorzystanie przycisku *Pokaż tabele* z paska narzędzi). Przeciągamy nazwę kolumny `PRACOWNICY.NrPracownika` na nazwę kolumny `POKOJE.NrPracownika`. Po pojawieniu się okna zatytułowanego *Edytowanie relacji* zaznaczamy opcję *Wymuszaj więzy integralności*. Zwróćmy uwagę na to, że w tym przypadku domyślnym typem złączenia jest „jeden do jednego”. Zatwierdzamy utworzenie złączenia, klikając przycisk *OK* (rysunek 14.2).



Rysunek 14.2.

Ważne jest ustanowienie unikalnego indeksu na kolumnie `POKOJE.NrKlienta` przed zdefiniowaniem złączenia.

Należy również zdawać sobie sprawę z tego, że pomimo istnienia związku pomiędzy dwoma kluczami głównymi, związek ten nie jest symetryczny. W tabeli zawierającej klucz główny złączenia (na przykład `PRACOWNICY`), mogą istnieć elementy, które nie będą występować w tabeli definiującej klucz obcy (na przykład `POKOJE`). Niedopuszczalna jest jednak sytuacja odwrotna (istnienie wartości klucza obcego, nie występujących w kluczu głównym złączenia).

Związki typu „wiele do wielu”

Związki typu „wiele do wielu” są stosunkowo powszechne. Ich reprezentacja w bazie danych może początkowo wydać się skomplikowana, lecz z chwilą, gdy opanujemy zasady, tworzenie tego typu związków okaże się bardzo łatwe.

Rozważmy związki pomiędzy klientami a pracownikami. Każdy klient może być obsługiwany przez różnych pracowników, każdy z pracowników natomiast ma do czynienia z wieloma klientami. Dlatego właśnie związki pomiędzy pracownikami a klientami są typu „wiele do wielu”. Kolejnym pytaniem, na które należy sobie odpowiedzieć, jest: „przez co warunkowane są powiązania pomiędzy klientami a pracownikami?”. Możliwe jest nawiązywanie kontaktów klientów z pracownikami bez dokonywania jakichkolwiek zakupów, lecz najprawdopodobniej tego typu kontakty nie mają znaczenia, przynajmniej z punktu

widzenia naszej bazy danych. Jedynym ważnym rodzajem kontaktów klientów z naszymi pracownikami jest kontakt związany z realizacją zamówień. Za każdym razem, gdy kupowany jest towar, zostaje złożone zamówienie. Zatem możemy stwierdzić, że powiązania pomiędzy klientami a pracownikami warunkowane są przez te zamówienia.

Zastanówmy się teraz nad powiązaniami pomiędzy klientami a zamówieniami. Są to związki typu „jeden do wielu". Tego samego typu związki istnieją pomiędzy pracownikami a zamówieniami.

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
3	Henryk	Bezbożny
4	Janina	Kozłowska

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Dostawca	Cena	Towar
1	*1*	*2*	Jan Strug	235,00 zł	Biurko
2	*4*	*1*	Fab. Mebli	234,00 zł	Krzesło
3	*1*	*3*	Jan Strug	415,00 zł	Stół
4	*2*	*4*	Błysk s.c.	350,00 zł	Lampa
5	*3*	*2*	Fab. Mebli	234,00 zł	Krzesło
6	*2*	*4*	Błysk s.c.	350,00 zł	Lampa
7	*2*	*2*	Jan Strug	235,00 zł	Biurko

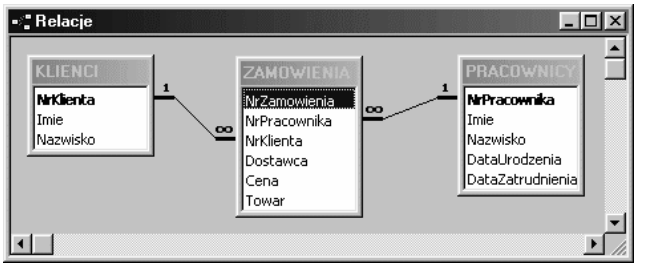
PRACOWNICY

NrPracownika	Imie	Nazwisko	DataUrodzenia	DataZatrudnienia
1	Jan	Kowalski	12 kwi 1966	01 maj 1999
2	Maria	Nowak	21 mar 1977	01 sty 2000
3	Zygmunt	Zdun	01 maj 1977	01 kwi 2002
4	Zofia	Kwiatkowska	03 kwi 1966	01 kwi 2002
5	Stefan	Czarnecki	12 kwi 1966	01 kwi 2002
6	Władysław	Mały	01 maj 1967	01 maj 2004

Być może wyda się to nieprawdopodobne, lecz dla utworzenia pary związków typu „jeden do wielu" w taki sposób, jaki opisaliśmy powyżej, wystarczy, aby powstał związek typu „wiele do wielu". W naszym przypadku, niejako samoistnie, powstał związek typu „wiele do wielu" pomiędzy tabelami KLIENCI oraz PRACOWNICY. Tak naprawdę nie istnieje w systemach relacyjnych baz danych żaden dodatkowy mechanizm, definiujący związki „wiele do wielu". Zawsze w przypadku konieczności utworzenia takiego związku posługujemy się dwoma związkami typu „jeden do wielu".

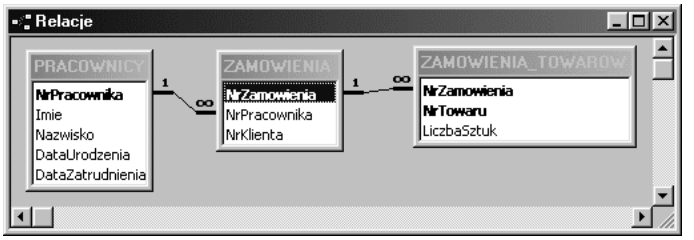
Istnieje jednak wymóg utworzenia tych powiązań w taki sposób, jak zostało to przedstawione na rysunku 14.3.

Rysunek 14.3.



Dowolne dwa związki typu „jeden do wielu", obejmujące trzy tabele, nie oznaczają jeszcze związku „wiele do wielu", czego przykładem może być sytuacja przedstawiona na rysunku 14.4.

Rysunek 14.4.



Z mojego doświadczenia wynika, że dość często można znaleźć tabelę, która będzie pomocna w celu utworzenia związku typu „wiele do wielu". W naszym przykładzie stała się nią tabela `ZAMOWIENIA`. Bywają jednak sytuacje, gdy taka tabela nie istnieje. Warto o tym wspomnieć, aby przybliżyć czytelnikowi problemy, które dość często mogą wystąpić w pracach nad rzeczywistymi bazami danych. W celu ilustracji posłużymy się ponownie naszą bazą danych. Dokonamy na niej kolejnych modyfikacji, zwiększając nieco poziom jej realizmu. Nasz przykład wyjaśni również, w jakich sytuacjach niezbędny może okazać się klucz główny zbudowany na dwóch kolumnach.

Konieczność zastosowania kluczy głównych zbudowanych na kilku kolumnach

Rozpatrzmy nasz przykład z rozdziału 12. Zidentyfikowaliśmy w nim sześć klas obiektów:

- klienci,
- towary,
- zamówienia,
- pracownicy,
- budynki,
- pokoje.

Skoncentrowaliśmy się wówczas na trzech tabelach:

- KLIENCI
- ZAMOWIENIA
- PRACOWNICY

Początkowo struktura tabeli ZAMOWIENIA była stosunkowo prosta, ponieważ wykorzystywaliśmy ją w celach demonstracyjnych. Teraz musimy przyjrzeć się bliżej strukturze tej tabeli, aby dostosować ją do naszych potrzeb. Przez chwilę będziemy posługiwać się przykładami z pliku *R14B.MDB*, następnie wykorzystamy zawartość pliku *R14C.MDB*.

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	NrTowaru
1	*1*	*2*	*1*
2	*4*	*1*	*3*
3	*1*	*3*	*4*
4	*2*	*4*	*2*
5	*3*	*2*	*3*
6	*2*	*4*	*2*
7	*2*	*2*	*1*

PRACOWNICY

NrPracownika	Imie	Nazwisko	DataUrodzenia	DataZatrudnienia
1	Jan	Kowalski	12 kwi 1966	01 maj 1999
2	Maria	Nowak	21 mar 1977	01 sty 2000
3	Zygmunt	Zdun	01 maj 1977	01 kwi 2002
4	Zofia	Kwiatkowska	03 kwi 1966	01 kwi 2002
5	Stefan	Czarnecki	12 kwi 1966	01 kwi 2002
6	Władysław	Mały	01 maj 1967	01 maj 2004

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
3	Henryk	Bezbożny
4	Janina	Kozłowska

TOWARY

NrTowaru	Dostawca	Cena	Towar
1	Jan Strug	235,00 zł	Biurko
2	Błysk s.c.	350,00 zł	Lampa
3	Fab. Mebli	234,00 zł	Krzesło
4	Jan Strug	415,00 zł	Stół

Jak widać, tabela ZAMOWIENIA składa się z kolumny ZAMOWIENIA.NrZamowienia, która stanowi klucz główny, oraz trzech kluczy obcych, odwołujących się do innych tabel.

Po przeanalizowaniu tej struktury można dojść do wniosku, że pozwala ona na realizowanie zamówień jednej tylko sztuki towaru. Na przykład zamówienie nr 3 zostało złożone przez Henryka Bezbożnyego i dotyczy zakupu stołu. Co moglibyśmy zrobić, gdyby pan Henryk zażyczył sobie do kompletu na przykład czterech krzeseł? W obecnej strukturze tabel nie ma możliwości zapisania kilku pozycji w ramach jednego zamówienia; nie ma nawet możliwości zapisania kilku sztuk jednego towaru. Z tego powodu każda sztuka towaru musi być zapisywana na osobnym zamówieniu, zatem zakup stołu i czterech krzeseł wymaga wpisania do bazy pięciu osobnych zamówień.

Istnieje kilka rozwiązań tego problemu. Dwa z nich wydają się logiczne i proste w implementacji, lecz mogą spowodować poważne problemy w bazie danych. Najlepszym rozwiązaniem będzie więc omówienie obu niebezpiecznych rozwiązań, aby zapobiec ich nieopatrznemu zastosowaniu.

Pierwsze, łatwe — ale błędne — rozwiązanie wygląda następująco:

BLEDNE_ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	Towar1	Towar2	Towar3	Towar4	Towar5
1	1	2	1	4			
2	4	1	3				
3	1	3	4	3	3	3	3
4	2	4	2	1	3		
5	3	2	3			2	
6	2	4	2		4		
7	2	2	1			2	

Wygląda nieźle, nieprawdaż? Zamiast pojedynczej kolumny przechowującej odwołania do towarów, mamy ich pięć. Henryk ma swoje cztery krzesła i stół, pozostali klienci mogą również kupować większe ilości sztuk poszczególnych towarów.

Mogłoby się wydawać, że to rozwiązanie jest całkiem dobre, lecz w praktyce jest zupełnie bezużyteczne. Po pierwsze, co stanie się, gdy liczba pozycji na zamówieniu będzie większa niż pięć? Możemy oczywiście dodać odpowiednią liczbę kolumn.

Załóżmy, że w ramach pojedynczego zamówienia maksymalna ilość sztuk wszystkich towarów będzie równa 30. W tym celu potrzebujemy 30 kolumn w tabeli. Problem polega na tym, że przeciętnie na zamówieniu występują trzy pozycje, więc średnio w każdym wierszu danych w tabeli marnujemy około 27 kolumn. To jest rzeczywiste marnotrawstwo miejsca na dysku i może znacznie spowolnić działanie bazy danych.

Dodatkowy problem może polegać na rozproszeniu informacji w ramach rekordu danych. Nie wiemy, która z kolumn zawiera informacje o krzesłach, więc w celu odnalezienia liczby krzeseł na zamówieniu musimy przeszukać zawartość całego wiersza danych. To jest stanowczo złe rozwiązanie.

Nieco lepszym, choć również niewystarczająco dobrym pomysłem, jest utworzenie następującej struktury:

BLEDNE_ZAMOWIENIA_2

NrZamowienia	NrPracownika	NrKlienta	Biurko	Lampa	Krzeslo	Stol
1	1	2	1			1
2	4	1			1	
3	1	3			4	1
4	2	4	1	1	1	
5	3	2		1	1	
6	2	4		1		1
7	2	2	1			

W tym podejściu każdy typ towaru jest reprezentowany przez jedną kolumnę w tabeli, a liczba, zapisana w polu kolumny określa ilość sztuk danego towaru, znajdującą się na zamówieniu. Henryk może kupić cztery krzesła i stół, a my wiemy dokładnie, gdzie szukać informacji na temat krzeseł. Nie musimy również martwić się o największą ilość sztuk towarów na zamówieniu, ponieważ w kolumnach odpowiadających danemu typowi towaru, możemy wpisać dowolną liczbę naturalną. Doskonale. Co jednak stanie się, gdy dodamy nowy typ towaru do naszego asortymentu? Będziemy musieli dodać nową kolumnę do tabeli ZAMOWIENIA. Wiele z firm posiada asortyment znacznie przekraczający dopuszczalną liczbę kolumn w większości systemów baz danych (najczęściej 255). Z tego względu takie rozwiązanie stanowczo nie nadaje się do zastosowania.

Nadszedł w końcu czas na zademonstrowanie właściwego rozwiązania. Wszystkie umieszczone poniżej tabele (aż do końca rozdziału) pochodzą z pliku *R14C.MDB*.

Najlepsze rozwiązanie naszego problemu okaże się bardzo proste, szczególnie dla osób przywykłych już do wykorzystywania wielu tabel. W przypadku zamówień i towarów mamy ponownie do czynienia ze związkiem typu „wiele do wielu". Skonstruujmy zatem związek „wiele do wielu" za pomocą dwóch związków „jeden do wielu" pomiędzy tabelami ZAMOWIENIA, TOWARY a dodatkową, trzecią tabelą. Tabela ta powinna znaleźć się w strukturze związku pomiędzy tabelami ZAMOWIENIA oraz TOWARY. Rozsądną dla niej nazwą może być ZAMOWIENIA_TOWAROW. Odpowiednie tabele będą wyglądały następująco:

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta	NrTowaru
1	*1*	*2*	*1*
2	*4*	*1*	*3*
3	*1*	*3*	*4*
4	*2*	*4*	*2*
5	*3*	*2*	*3*
6	*2*	*4*	*2*
7	*2*	*2*	*1*

ZAMOWIENIA_TOWAROW

NrZamowienia	NrTowaru	LiczbaSztuk
1	1	*1*
1	4	*1*
2	3	*1*
3	3	*4*
3	4	*1*
4	1	*1*
4	2	*1*
4	3	*1*
5	2	*1*
5	3	*1*
6	2	*1*
6	4	*1*
7	1	*1*

TOWARY

NrTowaru	Dostawca	Cena	Towar
1	Jan Strug	235,00 zł	Biurko
2	Błysk s.c.	350,00 zł	Lampa
3	Fab. Mebli	234,00 zł	Krzesło
4	Jan Strug	415,00 zł	Stół

Tabele te zawierają wszystkie informacje, które próbowaliśmy przechować za pomocą innych dwóch kiepskich rozwiązań. Oprócz tego ostatnie rozwiązanie nie jest obciążone problemami, które stanowiły integralny element w przypadku rozwiązań wcześniejszych.

Nie marnujemy miejsca w bazie danych, ponieważ wszystkie pola we wszystkich kolumnach są wypełnione.

Nie istnieje jakiekolwiek ograniczenie dotyczące liczby pozycji na zamówieniu czy też liczby różnych typów towarów w asortymencie. Dodanie nowej pozycji do zamówienia polega na dodaniu wiersza w tabeli ZAMOWIENIA_TOWAROW.

Jeśli wystąpi potrzeba dodania nowego typu towarów do bazy danych, wystarczy dopisać wiersz w tabeli TOWARY. Nie ma żadnej potrzeby ingerencji w struktury tabel.

Udało nam się w końcu udoskonalić strukturę naszej bazy danych. Jaki jednak ma to związek z wielokolumnowymi kluczami głównymi?

Kluczem głównym tabeli zamówień jest ZAMOWIENIA.NrZamowienia. Podobnie TOWARY.NrTowaru jest kluczem głównym tabeli towarów. Jednakże ani kolumna NrZamowienia, ani też NrTowaru nie może być kluczem głównym tabeli ZAMOWIENIA_TOWAROW, ponieważ w każdej z nich zupełnie poprawne będą wielokrotne wystąpienia tych samych wartości. Klucz główny można natomiast zbudować w oparciu o obie te kolumny. Dzięki temu nie ma problemu z powtarzaniem się wartości w polach każdej z tych kolumn z osobna.

ZAMOWIENIA_TOWAROW

NrZamowienia	NrTowaru	LiczbaSztuk
1	1	1
1	4	1
2	3	1
3	3	4
itd.	itd.	itd.

Niedopuszczalne jest jednak powtórzenie się par wartości w obydwu kolumnach, ponieważ stanowi to naruszenie zasady funkcjonowania klucza głównego.

ZAMOWIENIA_TOWAROW

NrZamowienia	NrTowaru	LiczbaSztuk
1	1	1
1	4	1
1	4	2
2	3	1
3	3	4
itd.	itd.	itd.

Taka sytuacja stanowi dość dobre odzwierciedlenie rzeczywistości, ponieważ z reguły na zamówieniu nie powinien pojawiać się kilka razy ten sam typ towaru. Do zapisu liczby sztuk jednego typu towaru w ramach zamówienia służy kolumna `LiczbaSztuk` w tabeli `ZAMOWIENIA_TOWAROW`.

Ogólne informacje na temat złączeń

Omówienie złączeń w związkach typu „wiele do wielu” zawierało dość sporo dodatkowych informacji. Warto więc zrobić małą przerwę i podsumować informacje, które pojawiły się do tej pory.

Wykorzystywanie wierszy zamiast kolumn zwiększa elastyczność

Omówione powyżej rozwiązanie problemu umieszczania kilku pozycji na zamówieniu jest bardzo elastyczne. Możemy swobodnie dodawać do zamówień pozycje, obejmujące dowolną liczbę sztuk danego towaru. Możemy również dodawać nowe towary do bazy danych. Obie te możliwości są dostępne bez konieczności wprowadzania jakichkolwiek dalszych modyfikacji w samej strukturze bazy danych.

Kilkakrotnie wspominałem już, że nowoczesne systemy zarządzania bazami danych (do których należy także MS Access) umożliwiają modyfikację struktury baz danych w sposób dość swobodny i nieskomplikowany. Możliwość taka jest bardzo ważna w procesie definicji struktury bazy danych, lecz w okresie wykorzystania bazy do codziennej pracy zmiany jej struktury powinny być ograniczone do naprawdę wyjątkowych sytuacji. Pamiętajmy, że w oparciu o tabele zbudowaliśmy formularze, zapytania i raporty. Każda zmiana struktury tabel wymaga sprawdzenia poprawności działania pozostałych

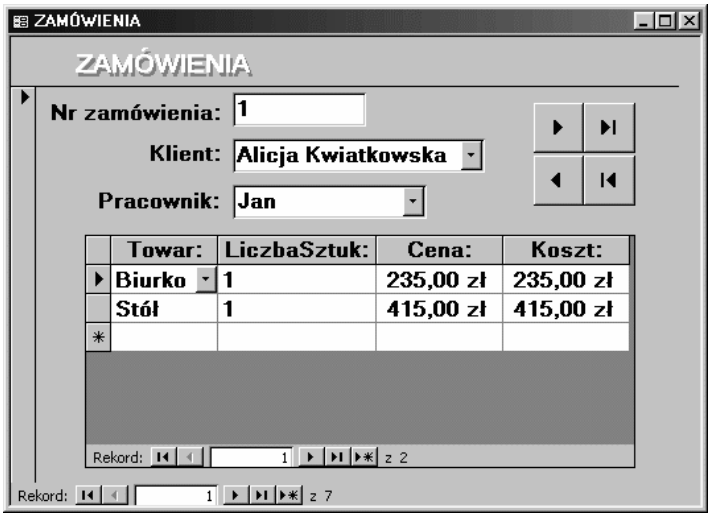
elementów bazy. Sytuacja, w której zmiana w strukturze tabel pociąga za sobą konieczność wprowadzenia zmian w którymś z pozostałych elementów bazy, jest bardzo prawdopodobna. Można więc wysnuć wniosek, że każda konstrukcja bazy danych, która wymusza częste dokonywanie zmian w strukturze tabel, jest bardzo mało funkcjonalna. Problem nie tylko w tym, że aplikacja taka jest dość kłopotliwa w utrzymaniu, lecz nasuwa bardzo prawdopodobne przypuszczenie, że popełniono jakiś błąd lub przeoczenie w trakcie definicji struktury bazy danych.

Wykorzystanie interfejsu graficznego

Złączenia wykorzystują liczby, zarówno w kluczach głównych, jak i w kluczach obcych. Liczby są wykorzystywane głównie z powodów praktycznych (bo tak jest po prostu najwygodniej), lecz bez problemu można wykorzystywać w tym celu napisy, czyli dane typu tekstowego. Należy jednak zdawać sobie sprawę z tego, że nie powinno mieć miejsca ręczne wypełnianie kolumn kluczy obcych w tabelach.

Weźmy ponownie naszą tabelę `ZAMOWIENIA_TOWAROW`. W celu wypełnienia zamówień wykorzystamy wygodny i estetyczny *GUI – Graphic User Interface* (*Graficzny interfejs użytkownika*). Dzięki temu będziemy mieli możliwość wyboru klienta z jednej listy rozwijalnej, pracownika z drugiej, natomiast towaru z trzeciej listy (rysunek 14.5).

Rysunek 14.5.



W ramach formularza użytkownik wprowadza wyłącznie niezbędne informacje. Cała praca związana z uzupełnianiem kluczy obcych, wykonywana jest przez system zarządzania bazą danych.

Omówienie sposobów tworzenia takich aplikacji lub wręcz serwisów WWW, wykorzystywanych w charakterze formularzy, obsługujących bazy danych, wykracza znacznie poza zakres tej książki. Książka niniejsza omawia bowiem zagadnienia relacyjnych baz danych, nie zaś sposoby wykorzystania aplikacji MS Access do tworzenia aplikacji wykorzystujących bazy danych. Osoby zainteresowane pewnymi standardowymi mechanizmami, dostępnymi w programie MS Access, mogą spróbować przeanalizować

przykładowy formularz o nazwie Zamowienia w pliku *R14C.MDB*. Formularz ten demonstruje podstawowy sposób tworzenia interfejsu użytkownika, zbudowanego w oparciu o kilka tabel, powiązanych pomiędzy sobą za pomocą różnych złączeń.

Następny rozdział zawiera jednak kilka ogólnych porad dotyczących tworzenia takich interfejsów, których głównym zadaniem jest odseparowanie użytkownika końcowego od abstrakcji relacyjnego modelu danych.

Mniej oczywiste klasy obiektów

Zasada projektowania tabel, przedstawiona przez nas we wcześniejszych rozdziałach, polegała na identyfikacji klas obiektów świata rzeczywistego. W tym rozdziale zdecydowałem się jednak na wprowadzenie tabeli ZAMOWIENIA_TOWAROW, którą raczej trudno wydedukować na podstawie analizy świata rzeczywistego. Na swoją obronę mam to, że uprzedzałem, iż jest to zaledwie ogólna zasada, od której mogą istnieć wyjątki. Listę tych wyjątków da się jednak przedstawić w sposób opisowy: otóż wyjątek mogą stanowić sytuacje występowania związków typu „wiele do wielu". W przypadku istnienia takiego związku przy jednoczesnym braku tabeli, która nadawałaby się do wykorzystania w charakterze „łącznika" tworzącego związek typu „wiele do wielu", najprawdopodobniej zaistnieje konieczność utworzenia mniej intuicyjnej tabeli, takiej, jaką są właśnie ZAMOWIENIA_TOWAROW.

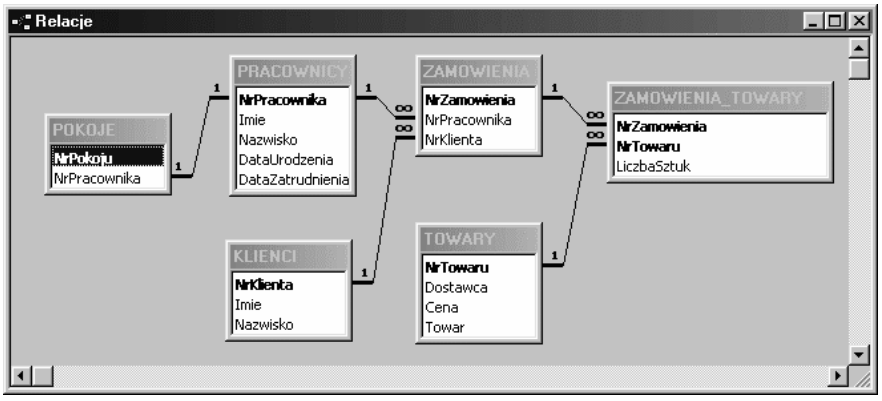
Terminologia

Jak mieliśmy okazję się przekonać, złączenia i klucze są proste w użyciu oraz niezwykle użyteczne. Udało się nam zatem postawić kolejny krok w stronę świata specjalistów baz danych. W tym świecie bardzo ważna jest umiejętność posługiwania się skomplikowaną terminologią, akceptowaną przez innych specjalistów baz danych. Jeśli nie zastosujemy się do tych zasad, zamiast należnego nam szacunku napotkamy mur niechęci jako intruzi. Dlatego ważna jest znajomość następujących pojęć: **rodzic**, **potomek**, **posiadanie**, **nadrzędny**, **podrzędny**, **zależność** czy też **klucz obcy**.

Na przykład tabela ZAMOWIENIA jest **w posiadaniu** tabela PRACOWNICY, dlatego jest jej **potomkiem**. Tabela ZAMOWIENIA jest **podrzędna** wobec tabeli KLIENCI, która jest jej **rodzicem** i również **posiada** tabelę ZAMOWIENIA, więc jest **nadrzędna** w stosunku do niej. Wszystkie tabele potomne posiadają przynajmniej jeden **klucz obcy**. Ponieważ tabela ZAMOWIENIA posiada dwie tabele **rodziców**, zawiera dwa **klucze obce** — NrPracownika oraz NrKlienta. **Klucze obce** ustalają **zależność** pomiędzy **rodzicem** a **potomkiem**. W tym przypadku mamy do czynienia z dwoma **kluczami obcymi**, więc istnieją dwie **zależności**.

Zwróćmy uwagę na to, że klucze obce tabeli ZAMOWIENIA tworzą związki z kluczami głównymi swoich rodziców. Jest to ważny wymóg, ponieważ tabele rodziców zawsze znajdują się po stronie „jeden" związku typu „jeden do wielu".

Rysunek 14.6 prezentuje związki pomiędzy tabelami z bieżącej wersji omawianej przez nas bazy danych.



Rysunek 14.6.

A oto zawartość wszystkich sześciu tabel bazy danych:

PRACOWNICY

NrPracownika	Imie	Nazwisko	DataUrodzenia	DataZatrudnienia
1	Jan	Kowalski	12 kwi 1966	01 maj 1999
2	Maria	Nowak	21 mar 1977	01 sty 2000
3	Zygmunt	Zdun	01 maj 1977	01 kwi 2002
4	Zofia	Kwiatkowska	03 kwi 1966	01 kwi 2002
5	Stefan	Czarnecki	12 kwi 1966	01 kwi 2002
6	Władysław	Mały	01 maj 1967	01 maj 2004

KLIENCI

NrKlienta	Imie	Nazwisko
1	Edward	Dziki
2	Alicja	Kwiatkowska
3	Henryk	Bezbożny
4	Janina	Kozłowska

ZAMOWIENIA

NrZamowienia	NrPracownika	NrKlienta
1	*1*	*2*
2	*4*	*1*
3	*1*	*3*
4	*2*	*4*
5	*3*	*2*
6	*2*	*4*
7	*2*	*2*

ZAMOWIENIA_TOWAROW

NrZamowienia	NrTowaru	LiczbaSztuk
1	1	1
1	4	1
2	3	1
3	3	4
3	4	1
4	1	1
4	2	1
4	3	1
5	2	1
5	3	1
6	2	1
6	4	1
7	1	1

TOWARY

NrTowaru	Dostawca	Cena	Towar
1	Jan Strug	235,00 zł	Biurko
2	Błysk s.c.	350,00 zł	Lampa
3	Fab. Mebli	234,00 zł	Krzesło
4	Jan Strug	415,00 zł	Stół

POKOJE

NrPokoju	NrPracownika
1	2
12	4
23	1
24	6

Czytelnikowi należy się małe wyjaśnienie. Nie powinienem robić sobie żartów z terminologii stosowanej przez specjalistów baz danych. W większości dziedzin życia konieczna jest jakaś forma werbalnego skrótu myślowego, używanego w celu usprawnienia komunikacji. Należy jednak zdawać sobie sprawę z tego, że terminologia taka powinna być stosowana wyłącznie w celu *usprawnienia* komunikacji, nie zaś w celu zabawy (z nowicjuszami) w kotka i myszkę, polegającej na niepotrzebnej komplikacji podstawowych zagadnień, które w gruncie rzeczy są proste i zrozumiałe. Zrozumienie podstaw jest tutaj kluczowe; po pokonaniu tego etapu cała skomplikowana terminologia jest wchłaniana przez nowego adepta w sposób nieomalże naturalny.



Zademonstrowane powyżej tabele nadal nie stanowią idealnego rozwiązania problemu, ponieważ tabela TOWARY wciąż zawiera nadmiarowe, powtarzające się dane (na przykład dotyczące dostawców). Jest to zabieg celowy, ponieważ te niedoskonałości posłużą jako ilustracja zabiegu usuwania nadmiarowych danych, którym zajmiemy się w rozdziale 15.