ĆWICZENIE NR 9

MODELE STRUKTUR BAZY DANYCH

Grzegorz Pająk

1. CEL ĆWICZENIA

Celem ćwiczenia jest zapoznanie z podstawami projektowania relacyjnych baz danych oraz zasadami funkcjonowania systemów zarządzania bazami danych na przykładzie programu Microsoft Access.

2. WPROWADZENIE

Baza danych jest to uporządkowany zbiór danych, dostępnych dla licznych użytkowników, w którym można przeprowadzić efektywne wyszukiwanie i aktualizowanie informacji. Bazy danych stanowią jeden z najszybciej rozwijających się kierunków współczesnej informatyki. Głównym powodem rozwoju są konkretne, codzienne potrzeby szerokiej gamy użytkowników, wśród których znajdują się szpitale, banki, administracja, instytucje zajmujące się transportem, nauką, edukacją, statystyką, itd.

Współcześni użytkownicy baz danych zainteresowani są szybkim, niezawodnym, możliwie tanim i łatwym dostępem do wiarygodnych informacji. Potrzeby te podyktowane są charakterem i naturą procesów społecznych i ekonomicznych występujących w rozwiniętych cywilizacyjnie społeczeństwach. Zastosowanie baz danych w biznesie to głównie przechowywanie danych personalnych oraz informacji dotyczących przeprowadzanych przez przedsiębiorstwo operacji. Menadżer, który ma do dyspozycji rozbudowany system baz danych może znacznie łatwiej otrzymać istotne informacje, dzięki którym

podejmuje szybsze decyzje oparte na szczegółowej analizie wielu informacji, niemożliwej do przeprowadzenia bez automatyzacji wielu procesów. Ze względu na specyficzne wymagania do podstawowych kryteriów oceny baz danych używanych w biznesie należy dokładność składowanych danych, łatwość dostępu, elastyczność i możliwość przystosowania do zróżnicowanych potrzeb poszczególnych szczebli zarządzania oraz bezpieczeństwo przechowywanych informacji. Ostatnie wymienione kryterium zyskuje coraz większe znaczenie w sytuacji, gdy wiele firm udostępnia część swoich zasobów w ogólnodostępnych sieciach komputerowych, a niektóre z nich posiadają wiele oddziałów pomiędzy którymi odbywa się transmisja poufnych danych.

Dane zbierane w bazach danych muszą zapewniać możliwość sprawnego wyszukiwania i dostarczania potrzebnych informacji. Nie mogą więc być one składowane w sposób przypadkowy. Ponadto ze względu na specyfikę sposobu przetwarzania informacji przez komputer istnieje konieczność wyróżnienia dwóch poziomów w bazie danych. Jeden z nich to poziom logiczny opisujący sposób w jaki bazę widzi użytkownik. Powinien on być skonstruowany tak, aby zapewnić maksymalnie naturalny sposób dostępu do informacji zawartej w bazie danych. Drugim poziomem jest poziom fizyczny - związany z samym komputerem i niewidoczny dla użytkownika. Powinien zapewniać organizację danych w sposób umożliwiający najszybszy dostęp do informacji, co oznacza konieczność strukturalizacji danych zoptymalizowaną pod kątem sposobu pracy komputera. W czasie ponad czterdziestu lat prac teoretycznych i praktycznych różnica pomiędzy tymi dwoma poziomami stawała się coraz bardziej wyraźna. Rozwój narzędzi do budowy baz danych doprowadził do sytuacji, w której zarówno użytkownik jak i projektant tego typu systemów zajmuje się jedynie poziomem logicznym, bez konieczności rozwiązywania problemów związanych z fizycznym poziomem bazy danych.

Istnieje kilka alternatywnych podejść do problemu struktury danych w bazie, które opisują tzw. model danych. Model określa metody strukturalizacji danych w sposób pozwalający na późniejsze znalezienie potrzebnych informacji i jednoczesne uniknięcie niepotrzebnych powtórzeń zwiększających rozmiar bazy danych. Jednym z najpopularniejszych obecnie modeli jest model relacyjny. Do jego dużej

popularności przyczyniła się prostota i solidne podstawy teoretyczne na których został oparty.

Podstawową formą organizacji danych w tym modelu relacja.

Definicja: Niech będą dane zbiory D_1, D_2,…,D_n. Relacją R na tych zbiorach nazywamy dowolny podzbiór ich iloczynu kartezjańskiego. Relację R zapisujemy:

$$R(D_1, D_2,\ldots,D_n),\ R \subset D_1 \times D_2 \times \ldots \times D_n.$$

Zapis postaci $R(D_1, D_2,\ldots,D_n)$ nazywany jest schematem relacji R, a elementy $D_1, D_2, \ldots, D_n$ atrybutami lub składnikami relacji. Przykładowym schematem relacji Pracownik jest:

Pracownik(PESEL, Nazwisko, Imię, Adres, Data urodzenia)

Dla danego schematu relacji ciąg wartości jego atrybutów jest nazywany krotką. Przykładową krotką dla relacji Pracownik jest: <70052000234, Jan, Kowalski, Zielona Góra ul. Podgórna 50, 1970-05-20>. Relację często definiuje się również jako zbiór takich krotek o formie $<d_1, d_2, \ldots, d_n>$, że $d_1 \in D_1$, $d_2 \in D_2$, …, $d_n \in D_n$.

Pojęciem istotnym dla relacyjnego modelu danych jest identyfikator kluczowy:

Definicja: Identyfikatorem relacji R o schemacie R(A,B,C,D,...) nazywamy składnik lub ciąg składników, których wartości określają w sposób jednoznaczny krotkę relacji. Identyfikator kluczowy (lub klucz) relacji, to jeden z jej identyfikatorów, dowolnie wybrany.

Klucz w schemacie relacji jest zaznaczany przez podkreślenie odpowiednich składników. Kluczem w relacji Pracownik może być atrybut PESEL, którego unikalność (a więc jednoznacznie określenie krotki) jest zagwarantowana dla dowolnego zbioru pracowników. Stąd schemat relacji Pracownik może być zapisany jako:

Pracownik(<u>PESEL</u>, Nazwisko, Imię, Adres, Data urodzenia)

Klucz może być jednym z atrybutów relacji, których obecność wynika z przeprowadzonej analizy problemu, klucz taki jest nazywany naturalnym. W wielu przypadkach w relacji nie występują jednak składniki, których cechy gwarantują unikalność wartości. W takiej sytuacji jako klucz

wprowadza się nowy składnik, którego wartości zagwarantują jednoznaczną identyfikacją krotek. Taki klucz jest nazywany sztucznym. W roli klucza sztucznego najczęściej występuje liczba porządkowa.

Duża elastyczność relacyjnych baz danych wynika z możliwości łatwego manipulowania danymi. Ta cecha modelu relacyjnego wiąże się z odpowiednim zaprojektowaniem struktury danych. Informacje o relacjach i ich atrybutach przekazywane przez użytkowników, na ogół nie pozwalają na ich bezpośrednie wykorzystanie. Projektant musi zaprojektować podział zbioru atrybutów na poszczególne schematy relacji w taki sposób, aby uzyskać pewne pożądane właściwości bazy danych. Nieprawidłowe zaprojektowanie schematów relacji przechowywanych w bazie jest przyczyną dublowania się danych, ich niespójności i anomalii podczas aktualizacji. Wszystkie te zjawiska uniemożliwiają lub co najmniej bardzo utrudniają poprawną eksploatację bazy danych.

W ramach projektowania schematów baz danych, najistotniejszą czynnością jest normalizacja relacji, czyli doprowadzenie relacji do odpowiedniej postaci normalnej [1], [2], [4]. Relacje opisane na etapie analizy przez użytkowników bazy nazywane są relacjami w formie dowolnej. W celu wprowadzenia pojęcia formy normalnej zostaną podane definicje zależności funkcjonalnej. W przykładach ilustrujących poniższe definicje została wykorzystana relacja postaci:

Pracownik(<u>PESEL</u>, Nazwisko, Imię, DataZ, OkresZ, Wydział, Kierownik)

gdzie: PESEL, Nazwisko, Imię, Adres – dane personalne pracownika; DataZ, OkresZ – informacje o zwolnieniach chorobowych (odpowiednio data i okres); Wydział – nazwa wydziału, na którym pracuje dana osoba; Kierownik – dane kierownika wydziału.

Definicja: Składnik B jest funkcjonalnie zależny od składnika A w schemacie relacji R(A, B, C, D) jeżeli każdej wartości $a \in A$ jest przyporządkowana tylko jedna wartość $b \in B$. Zależność funkcjonalną B w stosunku do A zapisujemy $A \rightarrow B$.

Np. W relacji Pracownik:

- Zależność PESEL→Nazwisko jest zależnością funkcjonalną ponieważ każdemu numerowi PESEL jest przyporządkowane dokładnie jedno nazwisko (w przypadku osób o podwójnym nazwisku obydwa zapisywane są w atrybucie Nazwisko).
- Zależność PESEL→DataZ nie jest zależnością funkcjonalną, ponieważ jednemu numerowi PESEL może być przyporządkowanych wiele dat zwolnienia, jeżeli pracownik przebywał na zwolnieniu kilkakrotnie.

Definicja: Składnik B jest w zależności funkcjonalnej elementarnej od składnika A w schemacie relacji R(A, B, C, D) jeżeli jest funkcjonalnie zależny od A i nie jest funkcjonalnie zależny od części A.

Np. W relacji Pracownik:

- Zależność PESEL, DataZ→OkresZ jest zależnością funkcjonalną elementarną, ponieważ okres zwolnienia nie jest funkcjonalnie zależny tylko od numeru PESEL (jeden pracownik mógł być na zwolnieniu kilkakrotnie) i nie jest zależny funkcjonalnie tylko od daty zwolnienia (w tym samym dniu mogło pójść na zwolnienie kilku pracowników).
- Zależność PESEL, DataZ→Nazwisko nie jest zależnością funkcjonalną elementarną, ponieważ nazwisko jest zależne funkcjonalnie od samego numeru PESEL.

Definicja: Składnik B jest w zależności funkcjonalnej bezpośredniej od składnika A w schemacie relacji R(A, B, C, D) jeżeli nie istnieje taki składnik C dla którego: A→C i C→B.

Np. W relacji Pracownik:

- Zależność PESEL→Wydział jest zależnością funkcjonalną bezpośrednią, ponieważ nie istnieje składnik, który jednocześnie jest zależny od numeru PESEL i od którego jest zależna nazwa wydziału.
- Zależność PESEL→Kierownik nie jest zależnością bezpośrednią, ponieważ składnik Wydział jest zależny funkcjonalnie od numeru PESEL (każdy pracownik pracuje na jednym wydziale) i

jednocześnie od tego składnika są zależne funkcjonalnie dane kierownika (każdy wydział ma jednego kierownika)

Znajomość definicji zależności funkcjonalnych pozwala na wprowadzenie pojęcia pierwszej, drugiej i trzeciej formy normalnej relacji (IFN, IIFN i IIIFN). Te formy normalne stanowią podstawę procesu normalizacji danych. W teorii relacyjnych baz danych jest wykorzystywana również czwarta i piąta postać normalna, jednak nie zostały one omówione w tym opracowaniu. Odpowiednie definicje można znaleźć w literaturze [1], [2], [4].

Definicja: Relacja R jest w pierwszej formie normalnej jeśli każdy ze składników, który nie jest elementem klucza, jest w zależności funkcjonalnej od klucza.

Np. Relacja Pracownik nie jest w IFN normalnej ponieważ data i okres zwolnienia chorobowego (składniki DataZ i OkresZ) nie są zależne funkcjonalnie od przyjętego klucza (numeru PESEL). Taka sytuacja wskazuje na błędnie określony klucz, który należy uzupełnić o składnik DataZ. Po takim zabiegu relacja będzie w pierwszej formie normalnej. Ostatecznie relacja przyjmuje postać

Pracownik(PESEL, Nazwisko, Imię, DataZ, OkresZ, Wydział, Kierownik)

Definicja: Relacja R jest w drugiej formie normalnej jeżeli jest IFN oraz każdy ze składników, który nie jest elementem klucza, jest w zależności funkcjonalnej elementarnej od klucza.

Np. Powyższa relacja Pracownik nie jest w IIFN, ponieważ istnieją składniki (Nazwisko, Imię, Wydział, Kierownik), które nie są zależne elementarnie od klucza. W celu doprowadzenia relacji do IIFN można dokonać dekompozycji na dwie relacje o schematach:

Pracownik1(PESEL, Nazwisko, Imię, Wydział, Kierownik)
Pracownik2(PESEL, DataZ, OkresZ)

Definicja: Relacja R jest w trzeciej formie normalnej jeżeli jest w IIFN oraz każdy ze składników, który nie jest elementem klucza, jest w zależności funkcjonalnej bezpośredniej od klucza.

Np. Przedstawiona powyżej relacja Pracownik2 jest relacją w IIIFN, natomiast Pracownik1 jest tylko w IIFN, ponieważ atrybut Kierownik nie

jest zależny funkcjonalnie bezpośrednio od klucza. Podobnie jak w przykładzie powyższym relacja ta może być doprowadzona do IIIFN po dekompozycji:

Pracownik1-1(PESEL, Nazwisko, Imię, Wydział)
Pracownik1-2(Wydział, Kierownik)

Po wykonanym procesie normalizacji dane zostają sprowadzone do optymalnej formy o strukturze umożliwiającej łatwą manipulację zawartością bazy, a ponadto najmniejszej możliwej objętości. Taka forma danych pozwala na stosunkowo prostą implementację bazy danych.

Stworzenie aplikacji bazodanowej wiąże się z wyborem odpowiedniego narzędzia. Rozwój informatyki doprowadził do stanu, w którym implementacja bazy danych jest możliwa bez znajomości języka programowania i sprowadza się do wykorzystania gotowych programów – tzw. systemów zarządzania bazami danych (SZBD). Są to aplikacje, które zawierają wiele typowych mechanizmów oraz narzędzi pozwalających na zbudowanie bazy danych o dowolnej strukturze. Jedynym ograniczeniem jest przyjęty w danym SZBD model danych. Ponieważ większość popularnych programów tego typu posługuje się modelem relacyjnym nie ma problemów z implementacją bazy, której struktura została stworzona zgodnie z powyższą procedurą.

3. PRZYKŁAD

Należy zaprojektować strukturę bazę danych dla potrzeb biblioteki. Biblioteka posiada na stanie książki. Każda książka ma unikalny numer (sygnaturę), jest opisana poprzez podanie autora i tytułu. W bazie należy przechowywać również dane wypożyczających (nazwiska, adresy). Każdy wypożyczający posiada swój unikalny numer nadawany w chwili zapisania do biblioteki. Wypożyczającym przypisano różne kategorie (np. student, wykładowca). Kategorie te wyznaczają termin zwrotu oraz maksymalną liczbę książek do wypożyczenia. Baza powinna zawierać informacje o wypożyczanych książkach, z datą wypożyczenia i datą zwrotu włącznie.

W wyniku analizy potrzeb personelu biblioteki zostały sformułowane następujące relacje w formie dowolnej:

Książki(IDKS, Tytuł, Autor),

Wypożyczający(IDW, Nazwisko, Adres, IDKT, NazwaKT, Termin, Ilość),

Wypożyczenia(IDW, Nazwisko, IDKS, Tytuł, DataW, DataZ).

Przyjęte oznaczenia:

IDKS	identyfikator książki (sygnatura),
Tytuł	tytuł książki,
Autor	autor książki,
IDW	identyfikator wypożyczającego,
Nazwisko	nazwisko wypożyczającego
Adres	adres wypożyczającego,
IDKT	identyfikator (symbol) kategorii nadanej wypożyczającemu,
NazwaKT	nazwa kategorii nadanej wypożyczającemu,
Termin	maksymalny termin wypożyczenia (wynikający z nadanej kategorii),
Ilość	maksymalna ilość wypożyczanych jednorazowo książek (wynikająca z nadanej kategorii),
DataW	data i czas wypożyczenia książki,
DataZ	data i czas zwrotu książki (pusta jeżeli książka nie jest zwrócona)

Zestaw przykładowych danych, którymi mogą być wypełnione powyższe relacje można przedstawić w postaci tabelarycznej:

IDKS	Tytuł	Autor
12001	Wprowadzenie do projektowania baz danych	W. Cellary, Z. Krolikowski
12002	Programowanie systemów baz danych	K. Walczak
12003	Bazy danych	M. Muraszkiewicz, H. Rybiński

IDW	Nazwisko	Adres	IDKT	NazwaKT	Termin	Ilość
2003	J. Kowalski	Zielona Góra	S	Student	30	5
2004	A. Nowak	Sulechów	S	Student	30	5
2005	T. Wiśniewski	Zielona Góra	W	Wykładowca	90	10

IDW	Nazwisko	IDKS	Tytuł	DataW	DataZ
2003	J. Kowalski	12002	Programowanie syst...	2000-10-01	2000-10-25
2003	J. Kowalski	12003	Bazy danych	2000-10-01	2000-10-20
2004	A. Nowak	12001	Wprowadzenie do...	2000-10-15	2000-11-10
2005	T. Wiśniewski	12003	Bazy danych	2000-10-22	2000-12-01
2004	A. Nowak	12002	Programowanie syst...	2000-11-01	2000-11-27

Przyjmując rozmiary dla poszczególnych atrybutów można określić rozmiar wszystkich relacji dla pewnego stanu bazy danych. Dla uproszczenia załóżmy, że wszystkie wartości przechowywane są w postaci znakowej, a ich rozmiary wynoszą odpowiednio: IDKS – 4, Tytuł - 100, Autor – 50, IDW – 4, Nazwisko – 25, Adres – 50, IDKT – 1, NazwaKT – 10, Termin – 2, Ilość – 2, DataW – 10, DataZ – 10. Załóżmy dodatkowo, że w bibliotece jest 1000 książek i 100 wypożyczających, a każdy z nich dotychczas wypożyczył 10 książek. Dla takich wartości rozmiary poszczególnych relacji wynoszą odpowiednio:

Książki: 1000 * (4+100+50) = 154000
Wypożyczający: 100 * (4+25+50+1+10+2+2) = 9400
Wypożyczenia: 100*10*(4+25+4+100+10+10) = 153000
Razem: 316400

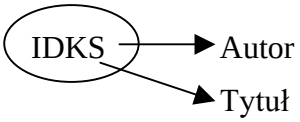
Na powyższym przykładzie można zauważyć redundancję przyjętego schematu relacji. Tytuły książek i nazwiska wypożyczających powtarzają się w relacji Wypożyczenia, dane wynikające z przypisanej kategorii w relacji Wypożyczający, itp. Sytuacja taka utrudnia posługiwanie się bazą danych i niepotrzebnie zwiększa jej rozmiar. Wszystkie te niedogodności zostaną usunięte w procesie normalizacji.

Przejście do IFN

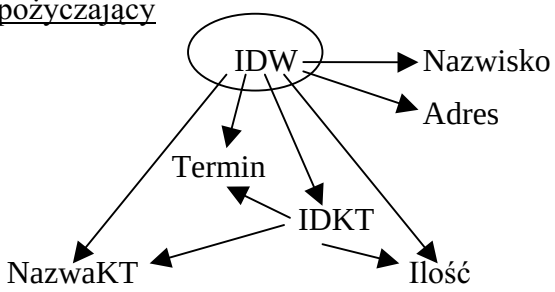
Sprawdzenie, czy relacja jest w IFN sprowadza się do sprawdzenia zależności funkcjonalnej wszystkich składników od klucza. W czasie analizy zadania pewne atrybuty zostały wstępnie zaznaczone jako klucze, jednak nie przeprowadzono dokładnej analizy. Do tego celu można

wykorzystać tzw. graf zależności funkcjonalnych. Jest to graf, którego wierzchołkami są nazwy atrybutów, a krawędziami strzałki oznaczające zależność pomiędzy odpowiednimi składnikami. Zakładając, że dana książka może być wypożyczana tylko raz dziennie grafy poszczególnych relacji mają postać:

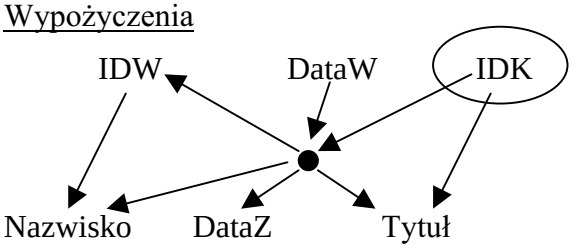
Książki



Wypożyczający



Wypożyczenia



Na podstawie kształtu powyższych grafów można stwierdzić, że w relacji Wypożyczenia nie wszystkie składniki są zależne funkcjonalnie od proponowanego klucza. Identyfikator wypożyczającego, nazwisko oraz data zwrotu książki nie są zależne jedynie od identyfikatora książki, ponieważ jedna książka jest wypożycza wiele razy, a więc jednej wartości IDKS odpowiada wiele daty i wypożyczających. Oznacza to konieczność rozbudowania klucza relacji Wypożyczenia, tak więc ostatecznie przyjmuje ona postać:

Wypożyczenia(IDW, Nazwisko, IDKS, Tytuł, DataW, DataZ).

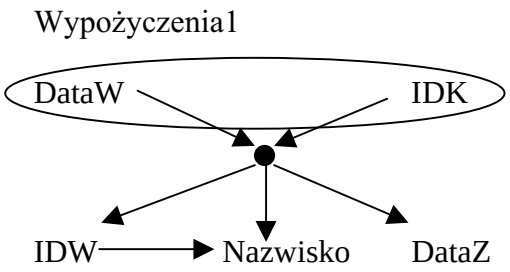
Przejście do II FN

Sprawdzenie czy relacja będąca w IFN jest w IIFN sprowadza się do ustalenia, czy wszystkie składniki są zależne funkcjonalnie elementarnie od klucza. W tym celu można wykorzystać powyższe grafy zależności funkcjonalnych. Relacja może nie być w IIFN tylko w przypadku, gdy jej klucz składa się z więcej niż jednego atrybutu. Eliminuje to z rozważań relację Książki i Wypożyczający. Analizując graf relacji Wypożyczenia można stwierdzić, że jeden składnik (Tytuł) zależy jedynie od części klucza. Oznacza to, że relacja nie jest w IIFN. W celu doprowadzenia jej do odpowiedniej formy normalnej należy dokonać dekompozycji na tyle relacji ile jest źródeł zależności funkcjonalnych elementarnych. Źródła te staną się kluczami nowopowstałych relacji. W ten sposób otrzymujemy:

Wypożyczenia1(IDW, IDKS, DataW, DataZ, Nazwisko)

Wypożyczenia2(IDKS, Tytuł)

Porównując nowe relacje z pozostałymi można zauważyć, że Wypożyczenia2 ma taki sam klucz jak Książki. Oznacza to możliwość scalenia odpowiednich relacji, co w tym przypadku sprowadza się do wykreślenia Wypożyczenia2, która w całości zawiera się w relacji Książki. Graf zależności funkcjonalnych relacji pozostałej po redukcji przedstawia rysunek:



Analizując powyższy graf można stwierdzić, że zostały wyeliminowane wszystkie zależności częściowe. Ostatecznie zestaw relacji w IIFN jest następujący:

Książki(IDKS, Tytuł, Autor),

Wypożyczający(IDW, Nazwisko, Adres, IDKT, NazwaKT, Termin, Ilość),

Wypożyczenia1(IDW, IDKS, DataW, DataZ, Nazwisko).

Przejście do IIIFN

Sprawdzenie czy relacja będąca w IIFN jest w IIIFN sprowadza się do stwierdzenia, czy wszystkie składniki są zależne funkcjonalnie bezpośrednio od klucza. Podobnie jak w poprzednich etapach normalizacji można to zrobić posługując się grafami zależności funkcjonalnych. Analizując powyższe grafy łatwo daje się zauważyć trzy zależności pośrednie w relacji Wypożyczający:

PESEL→IDKT→NazwaKT,

PESEL→IDKT→Termin,

PESEL→IDKT→Ilość

oraz jedną w relacji Wypożyczenia:

DataW, IDK→IDW→Nazwisko

Oznacza to, że relacje te nie są w IIIFN. W celu doprowadzenia ich do odpowiedniej formy normalnej należy dokonać dekompozycji na tyle relacji ile jest źródeł zależności funkcjonalnych bezpośrednich. Źródła te staną się kluczami nowopowstałych relacji. W ten sposób otrzymujemy:

Wypożyczający1(IDW, Nazwisko, Adres, IDKT)

Wypożyczający2(IDKT, NazwaKT, Termin, Ilość)

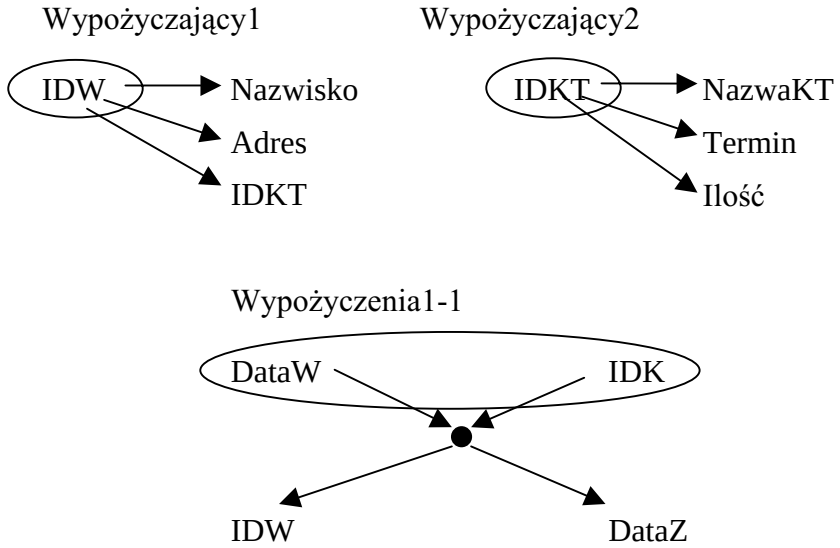
oraz

Wypożyczenia1-1(IDW, IDKS, DataW, DataZ).

Wypożyczenia1-2(IDW, Nazwisko).

Porównując nowopowstałe relacje z pozostałymi można stwierdzić, że klucz relacji Wypożyczenia1-2 jest taki sam jak relacji Wypożyczający. Oznacza to konieczność scalenia relacji, co w tym przypadku powoduje wykreślenie relacji Wypożyczenia1-2, która całkowicie zawiera się w

relacji Wypożyczający. Grafy zależności funkcjonalnych nowych relacji przedstawia rysunek:



Analiza powyższych grafów pozwala stwierdzić, że zostały wyeliminowane wszystkie zależności pośrednie, a więc relacje są w IIIFN. Ostatecznie zestaw relacji jest następujący:

Książki(<u>IDKS</u>, Tytuł, Autor),

Wypożyczający1(<u>IDW</u>, Nazwisko, Adres, IDKT)

Wypożyczający2(<u>IDKT</u>, NazwaKT, Termin, Ilość)

Wypożyczenia1-1(<u>IDW</u>, <u>IDKS</u>, DataW, DataZ).

Ostatnim etapem modelowania danych jest analiza trafności nazw relacji. Ponieważ nazwy powinny odzwierciedlać rzeczywistą zawartość relacji można zmodyfikować nazwę Wypożyczający2 na Kategorie, i przywrócić pierwotne nazwy relacjom Wypożyczający1 i Wypożyczenia1-1. Tak więc zestaw relacji przyjmuje postać:

Książki(IDKS, Tytuł, Autor),

Wypożyczający(IDW, Nazwisko, Adres, IDKT)

Kategorie(IDKT, NazwaKT, Termin, Ilość)

Wypożyczenia(IDW, IDKS, DataW, DataZ).

Korzyści wynikające z procesu normalizacji danych na tym etapie można najłatwiej wykazać sprawdzając rozmiar bazy danych. W celu porównania wielkości relacji w formie dowolnej i w IIIFN przyjmijmy założenia dotyczące rozmiaru składników i liczności relacji takie same jak na początku tego rozdziału. Dodatkowo należy określić liczność nowopowstałej relacji Kategorie. Ponieważ zarówno z przedstawionego opisu jak i z przykładowych danych wynika jedynie istnienie kategorii „Student" i „Wykładowca" przyjmujemy, że relacja ta ma dwie krotki. Stąd odpowiednie rozmiary wynoszą:

Książki:	1000 * (4+100+50) = 154000
Wypożyczający:	100 * (4+25+50+1) = 8000
Kategorie:	2 * (4+10+2+2) = 36
Wypożyczenia:	100*10*(4+4+10+10) = 28000
Razem:	**190036**

Aktualny rozmiar stanowi ok. 60% rozmiaru relacji w formie dowolnej. Należy ponadto uwzględnić poprawę tego stosunku wraz ze wzrostem bazy danych. Największe oszczędności wynikają z redukcji rozmiaru relacji Wypożyczenia (ok. 18% rozmiaru relacji w formie dowolnej), której wzrost jest najszybszy. Stąd oszczędności będą rosły wraz ze wzrostem liczby wypożyczeń.

Pozostałe korzyści wynikające z przeprowadzonej normalizacji, takie jak łatwiejsza aktualizacja danych, stają się widoczne dopiero na etapie budowy aplikacji przetwarzającej zaprojektowaną bazę danych. W tym celu w ćwiczeniu tym zaproponowano wykorzystanie systemu zarządzania bazami danych Microsoft Access. Wybór ten podyktowany był dużą popularnością pakietu Microsoft Office, w skład którego wchodzi program Access, oraz łatwym sposobem obsługi tego narzędzia, który nie wymaga znajomości zbyt wielu nieistotnych dla użytkownika szczegółów technicznych [3]. Zaprojektowane relacje implementuje się w programie Microsoft Access w formie tabel.

W celu zdefiniowania nowej tabeli, należy w oknie bazy danych kliknąć zakładkę **Tabele**, a następnie przycisk **Nowy**. Access wyświetla pole dialogowe z listą opcji: **Widok Arkusz**, **Widok Projekt**, **Kreator tabel, Importuj tabelę** oraz **Połącz tabelę**. W ćwiczeniu tym zostanie opisana jedynie metoda projektowania tabel przy pomocy opcji **Widok Projekt**, informacje o pozostałych opcjach można znaleźć w pliku pomocy oraz w literaturze [3].

Zaprojektowanie tabeli polega na podaniu listy pól, z jakich składać się będą wiersze tej tabeli. Pola deklaruje się w rubrykach górnej części okna (*Rys. 1*), podając dla każdego pola: **Nazwę pola**, **Typ danych** oraz **Opis**. Nazwa pola identyfikuje dane przechowywane w polu, typ określa rodzaj przechowywanych danych (np. tekst, liczba, data, rysunek), opis stanowi ułatwienie dla przyszłych użytkowników bazy (jest wyświetlany w pasku stanu podczas wprowadzania danych do pola). Microsoft Access dysponuje następującymi typami:

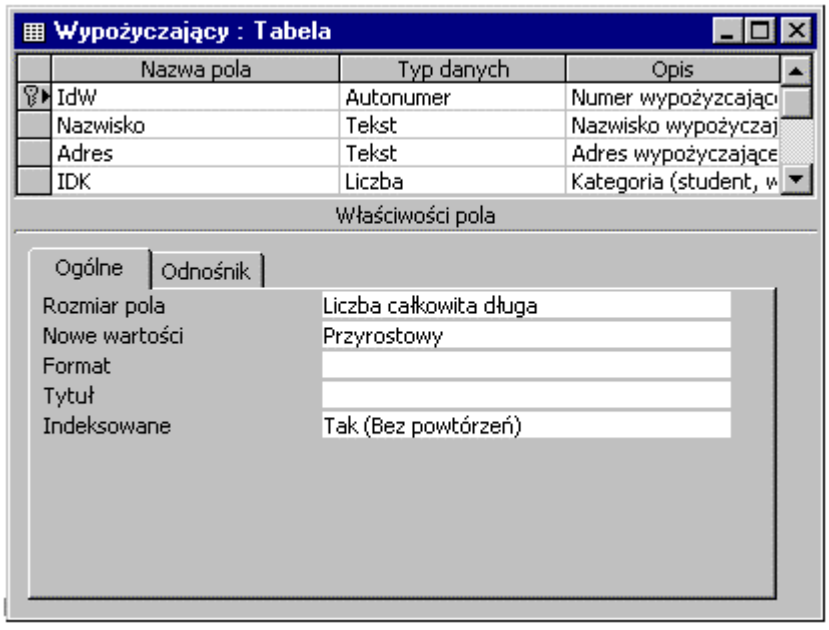
- Tekst – używany do przechowywania nazw, kombinacji nazw oraz liczb (np. adres z kodem pocztowym) lub liczb, które nie są używane we wzorach matematycznych (np. numer telefonu). Pozwala na umieszczanie w polach tekstów o długościach do 255 znaków.
- Memo – używany do zapisu rozbudowanej informacji tekstowej. W polu tego typu można zamieścić do 65 535 znaków.
- Liczba – używany do przechowywania liczb, na których będą przeprowadzane działania matematyczne (np. ilości, które będą sumowane).
- Data/Godzina – używany do przechowywania dat i godzin.
- Walutowy – używany do przechowywania liczb wyrażających kwoty w walucie.
- Autonumer – wykorzystywany do automatycznej numeracji kolejnych krotek liczbami naturalnymi, co pozwala na jednoznaczną ich identyfikację (np. w celu utworzenia klucza sztucznego).
- Tak/Nie – służy do przechowywania wartości logicznych typu prawda, fałsz.
- Obiekt OLE – umożliwia wstawianie do tabeli obiektów takich jak grafika, arkusz kalkulacyjny itp., które pochodzą z innych aplikacji.

- Hiperłącze – pozwala na przechowywanie w tabeli hiperłączy do pliku, określonego miejsca w pliku, strony HTML w sieci WWW lub strony HTML w intranecie, itp.
- Kreator odnośników – tworzy pole, które pozwala wybrać wartość z innej tabeli lub z listy wartości przy użyciu pola listy lub pola kombi.

Obok typu każde pole w tabeli posiada swoje cechy. Nadając cechy, określa się sposób, w jaki program Microsoft Access wyświetla i przetwarza dane przechowywane w polu. Cechy nadaje się polom w dolnej części okna projektu zatytułowanej **Właściwości pola**.

Po określeniu nazw oraz typów pól, należy jeszcze zadeklarować klucz podstawowy tabeli, zgodny z przyjętym w czasie projektowania struktury kluczem relacji. W tym celu należy ustawić podświetlenie w wierszu zawierającym pole kluczowe, a następnie wcisnąć przycisk paska narzędzi oznaczony kluczykiem. W wyniku jej operacji obok nazwy pola pojawi się symbol klucza.

Poniższy rysunek przedstawia okno projektu wraz ze strukturą zaprojektowanej tabeli „Wypożyczający”.



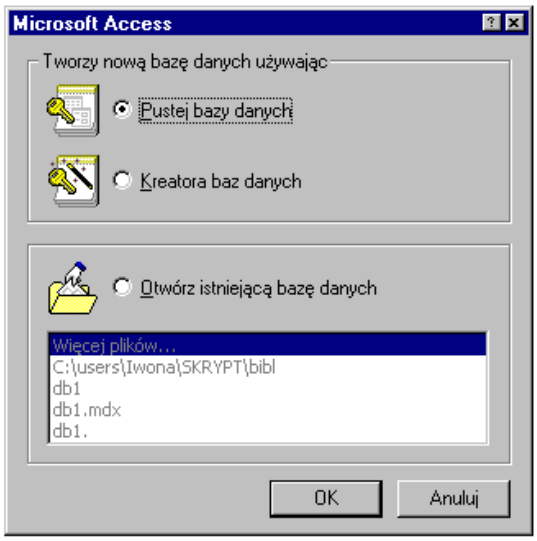
Rys. 1. Okno projektu tabeli

4. WYMAGANE PRZYGOTOWANIE DO ĆWICZEŃ

Przed przystąpieniem do wykonywania ćwiczenia należy zapoznać się z opisem programu Microsoft Access. Wymagana jest również znajomość podstawowych zasad pracy w systemie Microsoft Windows związanych z uruchomieniem i obsługą aplikacji.

5. URUCHOMIENIE PROGRAMU

Po uruchomieniu programu Microsoft Access pojawia się okno umożliwiające wybór sposobu budowy nowego projektu (*Rys. 2*).



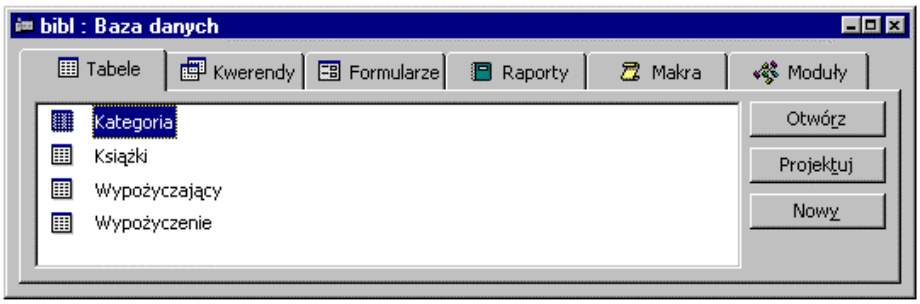
Rys. 2 Okno wyboru trybu tworzenia nowego projektu

Dostępne opcje: **Otwórz istniejącą bazę danych**, **Pusta baza danych** i **Kreator bazy danych**. Pierwsza z nich umożliwia odczytanie istniejącego projektu, natomiast dwie pozostałe utworzenie nowej bazy danych. Wybór kreatora powoduje wygenerowanie bazy w oparciu o jeden z kilku predefiniowanych szablonów. Utworzenie bazy w taki sposób jest bardzo proste i polega na wykonaniu kilku poleceń wyświetlanych przez kreator w formie kolejnych okien dialogowych. Tak uzyskana baza ma nieskomplikowaną strukturę i wykorzystuje jedynie w niewielkim stopniu możliwości programu Access. Należy ją traktować

jako szkielet projektu, przeznaczony do dalszej rozbudowy i adaptacji do indywidualnych potrzeb. Niezależnie od sposobu utworzenia każdy projekt bazy danych jest zapisywany na dysku w pojedynczym pliku o rozszerzeniu mdb.
Po odczytaniu istniejącej lub utworzeniu nowej bazy w obszarze roboczym programu pojawia się okno, które zawiera listę wszystkich elementów wchodzących w skład projektu. Elementy te są podzielone na kilka grup o różnym znaczeniu i zastosowaniu:

- **Tabele** – składowane dane,
- **Kwerendy** – elementy umożliwiające manipulowanie danymi,
- **Formularze** – elementy interfejsu użytkownika,
- **Raporty** – projekty wydruków,
- **Makra** – elementy automatyzujące czynności realizowane przez użytkownika,
- **Moduły** – zbiorem procedur napisanych w języku Access Basic.

Przykładowe okno bazy danych zostało pokazane na *Rys. 3*.



Rys. 3 Okno bazy danych

Każdy składnik bazy danych jest reprezentowany przez odpowiednią zakładkę. Na każdej zakładce znajduje się lista zawierająca nazwy elementów danego typu oraz przyciski umożliwiające manipulowanie jej zawartością. Zestaw przycisków jest różny i zależy od aktualnie wybranej zakładki. Możliwe są następujące polecenia:

- **Nowy –** umożliwia dodawanie nowych elementów
- **Otwórz –** ma różne działanie w zależności od typu elementów: otwiera tabelę w trybie edycji, uruchamia kwerendę i wyświetla jej wynik, lub wyświetla wskazany formularz,

- **Projektuj –** otwiera wybrany element w trybie modyfikacji struktury,
- **Podgląd** – występuje przy obiektach typu raport, wyświetla gotowy raport w trybie podglądu wydruku,
- **Uruchom** – występuje przy obiektach typu makro i moduł, uruchamia wybraną makrodefinicję lub modułu.

Opisane powyżej operacje uzupełniają opcje w menu **Plik** i **Edycja** oraz menu podręczne.

6. ZADANIA DO WYKONANIA

1. Dokonaj rozbudowy projektu „Biblioteka” o możliwość przypisywania każdej książce listy słów kluczowych, które pozwolą użytkownikom aplikacji na wyszukiwanie książek o wybranej tematyce. Przeprowadź proces normalizacji dla tak zmodyfikowanego problemu.
2. Zaprojektuj tabelę „Wypożyczający” zgodnie z uzyskanym schematem. Spróbuj uzupełnić właściwości poszczególnych pól. Sprawdź działanie reguły poprawności wprowadzając błędne dane tam, gdzie została ona określona. Utwórz odnośnik dla pola „IDK”.
3. Wypełnij przykładowymi danymi tabelę „Wypożyczający”.
4. Utwórz pozostałe tabele projektu „Biblioteka”, zgodnie ze schematem uzyskanym w wyniku normalizacji.
5. Na podstawie opisu programu Microsoft Access zapoznaj się z procesem tworzenia formularzy. Zaprojektuj formularze dla wszystkich stworzonych tabel.

LITERATURA

[1] Cellary W., Królikowski Z., Wprowadzenie do projektowania baz danych, WNT, Warszawa, 1988.

[2] Muraszkiewicz M., Rybiński H., Bazy danych, Akademicka Oficyna Wydawnicza RM, Warszawa, 1993.

[3] Pająk I., Pająk G., Łasiński K., Wprowadzenie do projektowania baz danych, Wydawnictwo Politechniki Zielonogórskiej, Zielona Góra, 1998.

[4] Ullman J.D., Systemy baz danych, WNT, Warszawa, 1988.