Piotr Ziniewicz

Program obsługi baz danych

MS Access

Wraz z rozwojem technik informacyjnych pojawiła się potrzeba magazynowania i sprawnego przetwarzania dużych ilości różnorakich danych. Sprawność systemów informacyjnych zależy w dużej mierze od czasu pozyskiwania z nich informacji oraz od ich zakresu. Podjęto próby standaryzowania struktur przechowujących dane jak również metod ich obróbki. W ten sposób w 1963 powstał termin „Baza danych".

Baza danych – jest to zbiór danych różnego typu (teksty, liczby, obrazy itp.) przechowywanych w specjalnie do tego celu przeznaczonych strukturach utworzonych z uwzględnieniem zasad przyjętego wcześniej **modelu danych**. Termin ten używany w języku potocznym obejmuje nie tylko sam zbiór danych, ale również środowisko komputerowe będące interfejsem dla użytkownika korzystającego z bazy danych.

Model bazy danych – jest to zbiór zasad, którymi należy się posługiwać podczas tworzenia bazy danych. W modelu danych określa się struktury oraz reguły, zgodnie z którymi są w nich umieszczane dane. Określany jest również zakres i sposób wykonywania operacji na danych tak, aby nie naruszyć ich integralności. Struktura danych jest definiowana poprzez specyfikację reprezentacji dozwolonych w modelu obiektów (encji) oraz ich związków. Najbardziej rozpowszechnione modele baz danych to:

- hierarchiczny model danych
- relacyjny model danych
- grafowy (sieciowy) model danych
- obiektowy model danych
- sieci semantyczne

Integralność danych – określenie sytuacji, w której dane zachowują postać (założoną w modelu bazy danych) podczas operacji takich jak odczyt, zapis, transmisja lub magazynowanie.

Encja – reprezentacja wyobrażonego lub rzeczywistego zbioru obiektów, posiadających taki sam zestaw cech opisujących każdy z tych obiektów a różniących się wartościami tych cech. Encja może reprezentować zarówno obiekt fizyczny jak i niematerialny. Encją może być zbiór osób (wraz z ich nazwiskami, imionami, numerami PESEL i adresami) jak również zbiór wizyt (data i godzina wizyty, rozpoznanie, zalecenia)

Bazy danych możemy podzielić według rodzajów struktur danych, które wykorzystują:

Bazy proste

- bazy kartotekowe
- hierarchiczne bazy danych

Bazy złożone

- bazy relacyjne
- bazy obiektowe
- bazy relacyjno-obiektowe
- strumieniowe bazy danych
- temporalne bazy danych

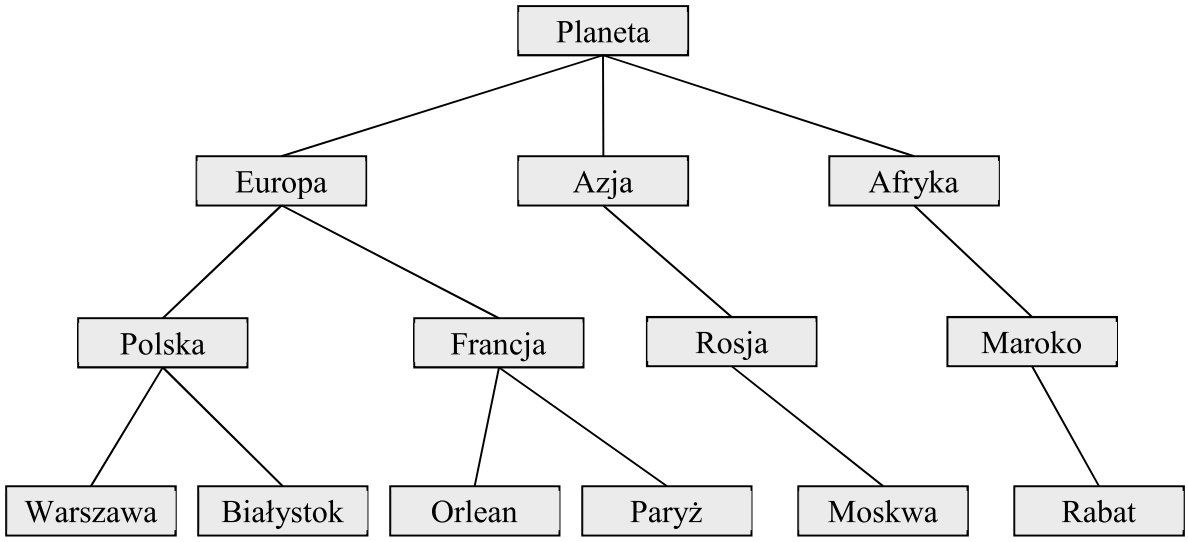
Najprostszym rodzajem bazy danych jest **baza kartotekowa**. Źródłem danych takiej bazy są samodzielne tablice, które nie mogą ze sobą współpracować ani być powiązane. Najczęściej jednak jest to po prostu pojedyncza tablica zawierająca informacje wykorzystywane tylko w jednym celu. Przykładem takiej bazy danych może być Książka Telefoniczna:

Nazwisko	Imię	Miasto	Ulica	Dom/lokal	Telefon
Kowalski	Jan	Warszawa	Praska	5/30	67825412
Łapiński	Cezary	Łapy	Pułaskiego	7	56471265
Zielińska	Anna	Kraków	Warszawska	8/19	75644125
……	……	……	……	……	……

Książka telefoniczna jako przykład bazy kartotekowej.

Tablica nazywana również tabelą składa się z kolumn (pól) oraz wierszy (rekordów). Każde pole opisuje jeden atrybut encji a każdy rekord odzwierciedla pojedynczy przypadek danej encji. W bazie takiej informacje można sortować, wyszukiwać lub narzucać filtry wyświetlające tylko pożądaną cześć danych. Jeżeli chodzi o reguły integralności danych można tu wspomnieć o narzuceniu obligatoryjności wypełnienia niektórych pól tabeli, bez których istnienie danego rekordu nie ma sensu (na przykład obligatoryjny byłby w tym przypadku numer telefonu, bez którego istnienie rekordu nie ma sensu). Aplikacje umożliwiające tworzenie tego typu baz oferują również możliwość stworzenia prostego interfejsu użytkownika umożliwiającego wyszukiwanie i inne operacje związane z dostępem do danych.

Kolejnym typem bazy danych są **bazy hierarchiczne**. W modelu hierarchicznym dane są przechowywane na zasadzie rekordów nadrzędnych-podrzędnych, tzn. rekordy przypominają strukturę drzewa. Każdy rekord (z wyjątkiem głównego) jest związany z dokładnie jednym rekordem nadrzędnym. Dane w takim modelu, są znajdowane na zasadzie wyszukiwania rekordów podrzędnych względem rekordu nadrzędnego. Przykładem takiego modelu może być struktura katalogów na dysku twardym komputera. System hierarchiczny budowany jest przeważnie w formie indukcyjnej tzn. dane są grupowane od ogółu do szczegółu. Oznacza to prostą formę wyszukiwania danych przechodząc poprzez kolejne poziomy szczegółowości. Dobrym przykładem takiej bazy będzie spis miejscowości na świecie.



Fragment spisu miejscowości zbudowanego na bazie modelu hierarchicznego.

Często wadą tego modelu jest brak możliwości budowania relacji pomiędzy rekordami różnych drzew. Warunki integralności danych są nieco bardziej złożone niż w przypadku bazy kartotekowej:

- Każdy rekord (z wyjątkiem pierwszego rodzica - korzenia drzewa) musi posiadać własnego, jednego rodzica

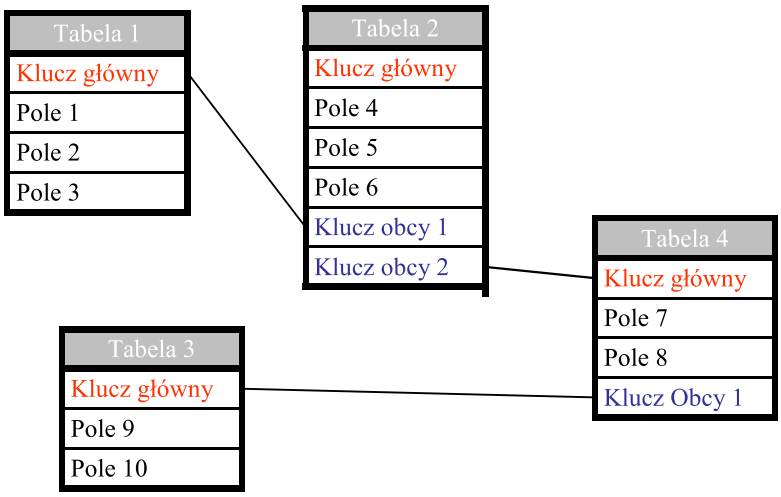
- Jeżeli usunięty zostaje dany rekord oznacza to, że usunięte zostają również wszystkie wywodzące się z niego rekordy – potomkowie (W przeciwnym wypadku doszło by do naruszenia pierwszej reguły – bezpośredni potomkowie usuniętego rekordu nie posiadali by rodzica).

Najbardziej rozpowszechnionym typem baz danych są **bazy relacyjne**. W bazach takich wiele tabel danych może współpracować ze sobą (są między sobą powiązane). Umożliwia to odzwierciedlenie relacji, w jakich znajdują się konkretne encje. Na przykład w przypadku biblioteki: czytelnicy, książki i wypożyczenia przechowywane będą w oddzielnych tabelach powiązanych ze sobą relacjami. Bazy relacyjne posiadają wewnętrzne języki programowania, wykorzystujące zwykle **SQL** do operowania na danych, za pomocą których tworzone są zaawansowane funkcje obsługi danych obejmujące ich filtrowanie, dodawanie, usuwanie i aktualizację. Relacyjne bazy danych podlegają następującym regułom integralności danych:

- Wszystkie wartości danych określone są typami danych. **Typem Danych** nazywamy opis rodzaju, struktury i zakresu wartości, jakie może przyjmować dane pole tabeli. Najczęściej spotykane typy danych to: *Tekst, Nota, Liczba, Data/Godzina, Walutowy, Autonumerowanie, Tak/Nie (Logiczny)*. Wiele typów danych posiada dodatkowe parametry określające zakres wartości. Na przykład pole typu *Tekst* posiada parametr długość określający maksymalną długość pola w znakach. Pole typu *Liczba* ma wiele podtypów określających zakres wielkości liczby oraz jej precyzji (ułamki).
- Wszystkie dane w bazie relacyjnej przedstawiane są w formie **dwuwymiarowych tabel**. Każda tabela zawiera zero lub więcej wierszy (rekordów) i jedną lub więcej kolumn (pól). Na każdy wiersz składają się jednakowo ułożone kolumny wypełnione wartościami które w każdym wierszu mogą być inne.
- Po wprowadzeniu danych do bazy, możliwe jest porównywanie wartości z różnych kolumn pochodzących z różnych tabel, i łączenie wierszy, gdy wartości z porównywanych kolumn są zgodne. Umożliwia to wiązanie danych i wykonywanie stosunkowo złożonych operacji w granicach całej bazy danych.
- Wszystkie operacje wykonywane są w oparciu o algebrę relacji, bez względu na położenie wiersza tabeli. Wiersze w relacyjnej bazie danych przechowywane są w porządku zupełnie dowolnym - nie musi on odzwierciedlać ani kolejności ich wprowadzania, ani kolejności ich przechowywania. Porządek danych można narzucić podczas ich wyświetlania poprzez odpowiednie posortowanie.
- Z braku możliwości identyfikacji wiersza przez jego pozycję pojawia się potrzeba obecności jednej lub więcej kolumn niepowtarzalnych w granicach całej tabeli, pozwalających odnaleźć konkretny wiersz. Kolumny te określa się jako **klucz główny** (ang. *primary key*) tabeli.

Klucz główny (*ang. Primary Key*). Oznacza wybrany zestaw kolumn tabeli (pól), jednoznacznie identyfikujący każdy wiersz tej tabeli (rekord). Kluczem głównym może być dowolny zestaw pól bądź jedno pole, ale często stosuje się rozwiązanie polegające na utworzeniu specjalnego pola, którego wartości domyślne pobierane są z sekwencji tak, aby zapewnić unikalność klucza. Typem tego pola jest najczęściej *Autonumerowanie* nadające polu wartości kolejnych liczb całkowitych. Warto zauważyć, że po usunięciu niektórych rekordów powstają „dziury” w numeracji. Nie ma to jednak najmniejszego znaczenia gdyż wartości klucza głównego nie mogą się powtarzać, ale nie muszą być wartościami kolejnymi.

Klucz obcy (*ang. Foreign Key*). Klucz obcy jest zbiorem pól tabeli (najczęściej pojedynczym polem) i przechowuje wartości klucza głównego innej lub tej samej tabeli. W ten sposób, poprzez porównanie wartości, można stwierdzić które rekordy pozostają w relacji.



Schemat relacyjnej bazy danych.

Powiązanie klucza obcego z kluczem głównym nazywamy relacją. Wyróżniamy 3 podstawowe typy relacji:

- **Relacja Jeden do Jednego** – każdemu rekordowi pierwszej tabeli odpowiada jeden i tylko jeden rekord drugiej tabeli. Relacja taka jest rzadko spotykana gdyż powiązane w niej rekordy dwóch tabel z powodzeniem można by umieścić w pojedynczej tabeli. W praktyce bywa ona stosowana, gdy ze względów bezpieczeństwa konieczne jest wyizolowanie pewnej części pól do osobnej tabeli i umieszczenie jej na serwerze o wyższym poziomie bezpieczeństwa. Przypadek, gdy całość danych jest umieszczana porcjami na różnych serwerach nazywamy Rozproszoną bazą danych.
- **Relacja Jeden do Wielu** – każdemu rekordowi pierwszej tabeli odpowiada wiele rekordów drugiej tabeli. Jest to najczęściej spotykany typ relacji. Dobrym przykładem jego zastosowania będzie baza danych, w której istnieje tabela z Pacjentami i tabela z Lekarzami rodzinnymi. Do jednego lekarza może być przypisanych wielu pacjentów co realizuje się przez umieszczenie w tabeli z pacjentami pola będącego kluczem obcym wskazującym na klucz główny tabeli z lekarzami.
- **Relacja Wiele do Wielu** – każdemu rekordowi pierwszej tabeli odpowiada wiele rekordów drugiej tabeli oraz każdemu rekordowi drugiej tabeli odpowiada wiele rekordów pierwszej tabeli. Przykładem zastosowania takiej relacji jest baza danych zawierająca informacje o wypożyczeniach książek z biblioteki. Dwie tabele (Wypożyczający i Książki) są powiązane tego typu relacją gdyż jeden wypożyczający może wypożyczać wiele książek a jedna książka mogła zostać wypożyczona wiele razy różnym wypożyczającym. W praktyce realizuje się tą relację dodając trzecią tabele pośredniczącą, w tym przypadku będzie to tabela Wypożyczenia. Umieszcza się w niej klucz obcy wskazujący na klucz główny tabeli Wypożyczający oraz drugi klucz obcy wskazujący na klucz główny tabeli Książki. Dodatkowo umieszcza się tam pola opisujące sam fakt wypożyczenia (np. data wypożyczenia, data zwrotu itp.). Tabela pośrednicząca pozostaje w relacji jeden do wielu z tabelą Wypożyczenia oraz z tabelą Książki zapewniając w ten sposób relację wiele do wielu pomiędzy tabelami Wypożyczenia i Książki.

SQL (*ang. **S**tructured **Q**uery **L**anguage*) – strukturalny język zapytań używany do tworzenia, modyfikowania baz danych oraz do umieszczania w niej danych i ich pobierania. Jest to język

zapytań opracowany w latach siedemdziesiątych w firmie IBM. Stał się on standardem w komunikacji z serwerami relacyjnych baz danych. Używanie tego języka polega na zadawaniu zapytań do bazy danych. Zapytania można podzielić na 3 główne grupy:

- **SQL DML** (*ang. Data Manipulation Language)* – Język Manipulacji Danymi
- **SQL DDL** (*ang. Data Definition Language)* – Język Definicji Danych
- **SQL DCL** (*ang. Data Control Language)* – Język Kontroli nad Danymi

Najważniejsze polecenia manipulacji danymi to:
SELECT – polecenie umożliwiające pobranie danych z bazy. Poniższy przykład przedstawia zapytanie wybierające wszystkie pola z tabeli pracownicy oraz tylko te rekordy w których pole *pobory* ma wartość większą od 2000. Wyświetlane dane będą sortowane malejąco według wartości pola *staz_pracy*

***SELECT * FROM** pracownicy **WHERE** pobory > 2000 **ORDER BY** staz_pracy **DESC**;*

INSERT – polecenie umożliwiające dopisywanie danych do bazy. Przykład demonstruje zapytanie dopisujące rekord do tabeli pracownicy oraz wypełniające odpowiednie pola podanymi wartościami

***INSERT INTO** pracownicy (imie, nazwisko, pobory, staz_pracy) **VALUES** ('Jan', 'Nowak', 5000, 2);*

UPDATE – polecenie umożliwiające aktualizację istniejących rekordów. Przykładowe zapytanie zwiększa o 50% pobory pracownikom, których staż pracy jest większy od 5.

***UPDATE** pracownicy **SET** pobory = pobory * 1.5 **WHERE** staz_pracy > 5;*

DELETE – polecenie umożliwiające usuwanie rekordów z bazy danych. Podane zapytanie usuwa z tabeli pracownicy wszystkie rekordy w których pole *imie* ma wartość Jan i pole *nazwisko* ma wartość Kowalski

***DELETE FROM** pracownicy **WHERE** imie = 'Jan' **AND** nazwisko = 'Kowalski';*

Najważniejsze polecenia definicji danych to:
CREATE (np. CREATE TABLE, CREATE DATABASE, ...) – utworzenie struktury (bazy, tabeli, indeksu, itp.). Przykładowe polecenie tworzy tabelę pracownicy o polach *imie* i *nazwisko* typu tekstowego o długości maksymalnej 255 znaków, *pobory* typu zmiennoprzecinkowego pojedynczej precyzji oraz *staz_pracy* typu liczba całkowita dwubajtowa.

***CREATE TABLE** pracownicy (imie varchar(255), nazwisko varchar(255), pobory float, staz_pracy int);*

DROP (np. DROP TABLE, DROP DATABASE, ...) – całkowite usunięcie struktury – tabeli bądź całej bazy danych. Podane polecenie usuwa tabele pracownicy

***DROP TABLE** pracownicy;*

ALTER (np. ALTER TABLE ADD COLUMN ...) – zmiana struktury (dodanie kolumny do tabeli, zmiana typu danych w kolumnie tabeli). Przykładowe polecenie dodaje do tabeli pracownicy pole *plec* typu tekstowego o długości jednego znaku.

***ALTER TABLE** pracownicy **ADD** plec varchar(1);*

Polecenia kontroli nad danymi umożliwiają nadawanie użytkownikom uprawnień do obiektów bazodanowych. Najważniejsze polecenie w tej grupie to:
GRANT – przyznawanie uprawnień do tabeli użytkownikowi
REVOKE – odebranie uprawnień praw do tabeli użytkownikowi