Bazy danych

Jacek Rumiński

Adam Bujnowski

wykład 2

Plan wykładu

1. Model warstwowy SBD
2. Modele danych – wprowadzenie
3. Model danych a struktury danych
4. Modele danych – model płaski

Bazy danych – model warstwowy

W informatyce bardzo często stosuje się **model warstwowy** w czasie projektowania i realizacji systemów (system – oznacza złożenie/grupę elementów, które można organizować na różne sposoby).
Cechą modelu warstwowego jest to, **że poszczególne warstwy mogą być** w pewien sposób **niezależne od siebie**. Współpraca pomiędzy nimi odbywa się poprzez uzgodnione protokoły (interfejsy). Oznacza to, że można w dowolnym czasie podmienić daną warstwę zachowując jedynie zgodność z protokołami (interfejsami) komunikacji pomiędzy warstwami. W ten sposób budowane są systemy operacyjne, implementacje sieci komputerowych, itd.

W większości przypadków podstawową warstwą (najniższą) jest warstwa fizyczna. Określa ona sposób realizacji zadań systemu na najniższym poziomie, tj. na poziomie implementacji sprzętowej. Inaczej warstwa fizyczna jest najdalej od bezpośredniej interakcji z użytkownikiem, czyli warstwy aplikacji.

Bazy danych – model warstwowy

Klasyczny wzorzec budowy systemu baz danych przedstawiono w normie ANSI-SPARC (American National Standards Institute, Standards Planning And Requirements Committee).:

- warstwa użytkownika (zewnętrzna) – jak dane widziane są przez użytkowników i aplikacje;
- warstwa koncepcyjna (pojęciowa) – logiczny widok danych (bez szczegółów dotyczących realizacji);
- warstwa fizyczna (wewnętrzna) – sposób przechowywania danych i dostępu do danych.

Bazy danych – model warstwowy

Model ANSI-SPARC (jak praktycznie każdy **model warstwowy**) **umożliwia ukrycie szczegółów implementacyjnych** (tj. fizycznej organizacji danych na nośniku danych) przed użytkownikiem. Warstwa fizyczna właściwie składa się z dwóch podwarstw: wewnętrznej reprezentacji danych dla potrzeb ich składowania na nośniku oraz z samego nośnika.
Wiąże się to z określonymi konsekwencjami:

- można zmienić nośnik, w żaden sposób nie wpływając na zmianę wewnętrznej organizacji danych;
- zmiana wewnętrznej organizacji danych w żaden sposób nie wpływa na aplikację (wyższe warstwy).

Jeden z twórców terminologii baz danych (szczególnie w zakresie relacyjnego modelu danych) Edgar Frank Codd zdefiniował 12 reguł (zwanych 12 przykazaniami, chociaż jest ich 13: od 0 do 12) jakie musi spełniać każdy system baz danych, aby mógł być nazwany relacyjnym.

Spośród 12 reguł warto w tym miejscu przytoczyć dwie z nich:

Reguła 8: Niezależność danych na poziomie fizycznym:

Zmiany na poziomie warstwy fizycznej (czyli jak dane są przechowywane, np. w tablicach, listach, itp.) nie mogą wymagać zmian w powiązanych z nią aplikacjach.

Reguła 9: Niezależność danych na poziomie logicznym:

Zmiany na poziomie warstwy logicznej (tabele, kolumny, wiersze, itp.) nie mogą wymagać zmian w powiązanych z nią aplikacjach.

Bazy danych – model warstwowy

Każdy dostawca DBMS stara się tak opracować wewnętrzny model organizacji danych w plikach, aby spełnić szereg wymagań funkcjonalnych (np. krótki czas zapisu danych, szybkie wyszukiwanie danych, przechowywanie duży zbiorów, itd.) oraz wymagań niefunkcjonalnych (np. zapewnić bezpieczeństwo danych).

Sposób realizacji warstwy fizycznej (ilość plików, ich powiązanie i budowa, itd.) oraz odwzorowanie warstwy logiczne (koncepcyjnej) na warstwę fizyczną (i odwrotnie) wpływa bezpośrednio na wydajność systemów zarządzania bazami danych (**„silników baz danych”**).

Plan wykładu

1. Model warstwowy SBD
2. Modele danych – wprowadzenie
3. Model danych a struktury danych
4. Modele danych – model płaski

Bazy danych – modele danych

W przypadku baz tradycyjnych zadania systemu zarządzania bazami danych pełnią ludzie (np. dodanie pozycji do bazy – wypisanie karty; usunięcie pozycji – podarcie karty; wyszukanie pozycji – odnalezienie karty, itp.).

Zastosowanie metod komputerowych do realizacji bazy katalogowej wymaga przede wszystkim przełożenia reprezentacji danych na papierze do formy elektronicznej. Jedną z najprostszych metod byłoby utworzenie tabeli w edytorze tekstu. Każdej kolumnie należałoby przypisać etykietę (nazwę opisującą rodzaj danych, przechowywanych w kolumnie). Kolejne wiersze stanowiłyby odpowiednik karty z bazy katalogowej.

Bazy danych – modele danych

Tabela (forma tradycyjnej bazy danych):

Sygnatura	Autor	Tytuł	Słowa klucze	ISBN
II0001	Kowalski Jan	Atlas anatomii człowiek	anatomia, człowiek	1234567890
II0002	Nowa Andrzej	Łagodne wprowadzenie do statystyki medycznej	statystyka, prawdopodobieństwo, analiza	2345678901
II0003	Kamieńska Anna	Tomografia komputerowa w praktyce	tomografia, CT, RTK	3456789012
II0004	Rudnik Hanna	Pediatria	pediatria, dzieci	4567890123
(...)	(...)	(...)	(...)	(...)
(...)	(...)	(...)	(...)	(...)
II9999	Fajowska Joanna	Dno oka	oko, siatkówka	6789012345

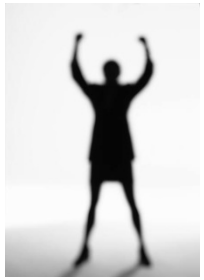
rekord

Tabela ma szereg własności. Przykładowe z nich to:

- jeden wiersz reprezentuje jedną kartę (czyli zapis/rekord danych dla konkretnej encji – tutaj książki), ale nic nie ogranicza nas przed duplikacją wierszy, czy wprowadzanie wierszy pustych,
- na przecięciu kolumny i wiersza występuje pole danych,
- pole danych może zawierać cokolwiek, zarówno pod względem treści jak i formy (np. zdjęcie),
- kolumna może być w pełni pusta, itp.

Bazy danych – modele danych

Duża dowolność budowy tabeli wynika z faktu, że jej twórcą, administratorem i jedynym użytkownikiem jest człowiek.



= SZBD

Trudno wyobrazić sobie uniwersalny algorytm, który przetwarzałby dane z tabeli, która ma charakter prezentacyjny (czyli ukazuje dane).

Pozostaje szereg ograniczeń, które chcielibyśmy wprowadzić, np. sygnatura nie może zawierać więcej niż 6 znaków, żadne pole nie może być puste, itd.
Wymagania takie oznaczają wykonywanie pewnych czynności, które sprawdzałyby poprawność wprowadzanych danych. Dla tabeli w formie prezentacyjnej czynności kontrolne musi przeprowadzać człowiek (lub specjalnie dedykowana i zrealizowana aplikacja).
Czy nie lepiej byłoby utworzyć takie uniwersalne struktury danych, aby uniwersalne oprogramowanie mogło sprawdzać poprawność wprowadzanych danych?

Bazy danych – modele danych

Rozpatrzmy również problem wyszukiwania danych.

1. Jak algorytmicznie wyszukać wszystkie te rekordy (wiersze), które spełniają pewne kryteria dla atrybutu (bądź grupy atrybutów)?

Praktycznie jedyne wyszukiwanie to szukanie na podstawie zgodność znaków (nie znajdziemy rekordu, tylko podobny tekst w obrębie tabeli).

Tabela w formie prezentacyjnej jest oczywiście bazą danych, ale stanowi jedynie proste przeniesienie danych z postaci papierowej do postaci elektronicznej. System zarządzania pozostaje taki sam – jest nim operator (człowiek).
Co należy zrobić, aby umożliwić zarządzanie danymi na poziomie programów?
Należy zastosować modele danych i ich realizacje programowe.

Bazy danych – modele danych

Model danych jest to zbiór zasad posługiwania się danymi:

- zbiór reguł określających strukturę danych (definicja danych),
- zbiór reguł określających operacje na danych (operowanie danymi),
-zbiór reguł określających poprawne stany bazy danych (integralność danych).

Zacznijmy od relacji model danych – struktury danych.

Plan wykładu

1. Model warstwowy SBD
2. Modele danych – wprowadzenie
3. Model danych a struktury danych
4. Modele danych – model płaski

Bazy danych – struktury danych

Załóżmy, że pragniemy przechować informacje o autorze książki. Podstawowe informacje to imię i nazwisko autora. Stosując podstawowe typy danych (w danym języku programowania), moglibyśmy przechować godność autora jako:

//typ_danych nazwa_zmiennej
tekst(60) imie_i_nazwisko;

gdzie *tekst* oznacza prosty typ danych, zbiór/tablicę do 60 znaków. Wprowadzanie do bazy opisu każdej książki odbywałoby się poprzez przypisanie (np. z użyciem funkcji strcpy w języku C) wartości do nazwy zmiennej i utrwaleniu zmiennej w strukturze fizycznej na nośniku (począwszy od jakiegoś adresu zbiór bajtów).

Przykładowo:

```
imie_i_nazwisko="Jan Kowalski";
zapiszDoPliku(imie_i_nazwisko);
imie_i_nazwisko="Jan Nowak";
zapiszDoPliku(imie_i_nazwisko);
```

Warto jednak zauważyć, że nie wiadomo (tj. bez oceny człowieka) gdzie w zapisanym tekście jest imię, a gdzie nazwisko.

Jak wówczas posortować zbiór książek według nazwiska autora?

W tym przypadku lepiej zastosować złożony typ danych:

```
osoba {
        teskt(40) nazwisko;
        tekst(20) imie;
}
```

Zastosowanie takiego typu do przechowania informacji o autorze mogłoby wyglądać następująco:

```
//utwórz zmienną typu osoba o nazwie autor
osoba autor;
autor.nazwisko=”Kowalski”;
autor.imie=”Jan”;
zapiszDoPliku(autor);
```

Projektowanie struktur danych jako zbiór atrybutów wymaga dwóch rzeczy:

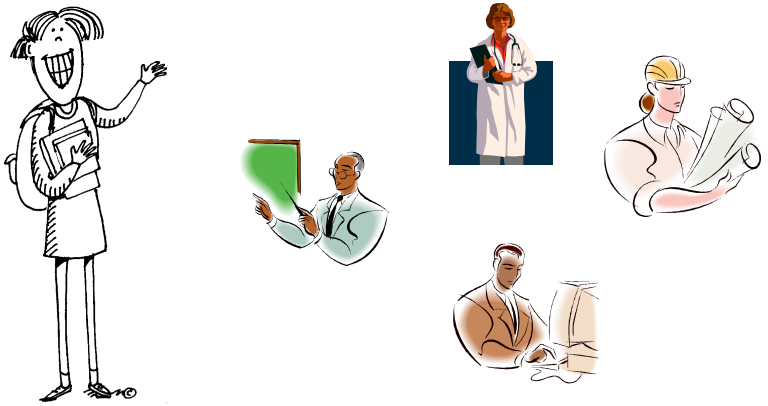
- znajomości prostych typów danych (na poziomie oprogramowania: języka programowania lub sytemu zarządzania bazą danych),
- zidentyfikowania wymagań dotyczących atrybutów zbioru encji.

Proste typy danych wprowadzane są przez języki programowania, normy i różne programy komputerowe, w tym DBMS. Każdy typ danych ma przypisaną etykietę, poprzez którą rozumie się najczęściej rodzaj i zakres reprezentacji danego typu (np. **int** (**integer**) to liczba całkowita 4 bajtowa, czyli o zakresie od 0 do 4294967295, lub od -2147483648 do 2147483647). Katalog typów danych jest określany w normie, w specyfikacji języka programowania lub w dokumentacji oprogramowania.

Bazy danych – struktury danych

Na czym polega identyfikacja wymagań dotyczących atrybutów zbioru encji?

Co to jest zbiór encji?



Wyobraźmy sobie studenta, który po raz pierwszy krąży po budynku wybranego wydziału. Obserwuje otaczający go świat i postrzega poszczególne byty podobne do siebie. Na czym polega podobieństwo? Identyfikuje zbiór atrybutów, które są właściwe podobnym bytom. Przykładowo wszyscy mają wzrost, nazwisko, imię, tytuł naukowy, poważny wygląd. Nasz student-wędrowca nazywa ten rodzaj bytu – wykładowcą. Zbiór encji (lub encję traktowaną jako zbiór) to zbiór wykładowców. Każda encja „wykładowca” ze zbioru encji charakteryzuje się właściwymi wartościami atrybutów.

Bazy danych – struktury danych

Student-wędrowca utworzył sobie strukturę danych (formę), opisującą wykładowców:

wykladowca {
liczba id;
teskt(40) nazwisko;
tekst(20) imie;
teskt(20) tytul;
liczba rokRozpoczeciaPracy;
}

Każdy konkretny wykładowca (encja, byt), reprezentowany będzie w formie rekordu (zbioru wartości poszczególnych pól struktury). Zbiór rekordów reprezentuje w systemie komputerowym zbiór encji w świecie rzeczywistym. Natomiast z wszystkich możliwych atrybutów (np. ilość włosów, długość palca serdecznego lewej ręki, nazwisko, itd.) wybiera te, które będą istotne w świetle obranego celu, zgodnie z którym, tworzona jest baza danych (np. system oceny wykładowców).

Bazy danych – struktury danych

Rozpatrzmy teraz przykładowe realizacje struktury danych "wykladowca":

Pierwsza demonstracja ukazuje kod źródłowy programu w języku programowania C, umożliwiający utworzenie struktury danych i przykładowych rekordów, a także realizujące podstawowe operacje zapisu/odczytu danych.

```
wykladowca {
        liczba id;
        teskt(40) nazwisko;
        tekst(20) imie;
        teskt(20) tytul;
        liczba rokRozpoczeciaPracy;
}
```

DEMO 2
Proste struktury danych w języku C

Bazy danych – struktury danych


```
//(…)
struct teacher{                         //struktura opisująca nauczyciela
        int id;                         //identyfikator
        char lName[40];                 //nazwisko
        char fName[20];                 //imię
        char title[20];                 //tytuł,/stopień
        int teachingStartYear;          //rok rozpoczęcia pracy
 };
struct teacher currentStaff[2];         //tablica do przechowywania rekordów
struct teacher allStaff[2];             //tablica rekordów odczytanych z pliku

currentStaff[0].id=1;                   //ale można wpisać dowolną liczbę całkowitą!
strcpy(currentStaff[0].lName,"Ruminski");
strcpy(currentStaff[0].fName,"Jacek");
strcpy(currentStaff[0].title,"dr inz.");
currentStaff[0].teachingStartYear=1994;

//(…)
//zapisz tablicę z rekordami do pliku "teachers.db"
//czytaj tablicę z rekordami z pliku …
```

Program przykładowy w języku C demonstruje ważne dwa aspekty związane z bazami danych:

- warstwę logiczną: struktura rekordów „**struct teacher**”;
- warstwę fizyczną: układ bajtów w tablicy (organizacja wewnętrzna) zapisanej w pliku „**teachers.db**”.

Zainteresowani mogą otworzyć plik „teachers.db” za pomocą edytora danych binarnych i dokonać analizy znaczenia i kolejności zapisanych tam bajtów. Pomocą może być dokumentacja języka C. Warto zwrócić uwagę na fakt, że wewnętrzna organizacja danych w pliku/plikach jest bardzo istotna ze względu na efektywność (wydajność, zdolność do rozszerzeń) korzystania z zapisanych danych. Z tych względów dostawcy DBMS starają się opracować własne, zoptymalizowane rozwiązania wewnętrznych struktur danych.

Plan wykładu

1. Model warstwowy SBD
2. Modele danych – wprowadzenie
3. Model danych a struktury danych
4. Modele danych – model płaski

Przypominając sobie definicję modelu danych okazuje się, że właśnie wypełniliśmy podstawowe wymagania co do modelu danych.
Model, które wykorzystaliśmy (przykład w języku C) to tak zwany **model płaski (ang. flat files)** lub prosty. Dlaczego płaski?

Otóż w naszym kodzie zdefiniowaliśmy strukturę jako zbiór atrybutów opisowych. Każdy rekord to zbiór wartości, zgodnie z wymaganiami struktury. Naszą bazą danych jest kolekcja rekordów, które nie zależą od siebie. **Żaden rekord nie jest powiązany (z uwagi na definicję struktury!) z innym rekordem**.
W przykładzie mieliśmy tylko 2 rekordy, ale równie dobrze moglibyśmy utworzyć ich 1000. W strukturze danych nie było definicji związku pomiędzy rekordami, stąd rekordy te są strukturalnie niezależne.

Bazy danych – struktury danych

Warto zauważyć, że kolekcja rekordów może być bardzo łatwo odwzorowana na postać prezentacyjną:

- nazwy kolumn <-> nazwy atrybutów w strukturze (lub powiązane etykiety tekstowe),
- wartości pól w wierszu <-> wartości atrybutów z rekordu.

Liczba wierszy odpowiada wówczas liczbie rekordów.

typ rekordu Pracownicy:

```
{
  numer PESEL;          <- typ atrybutu
  tekst(30) Nazwisko;
  tekst(20) Imię;
  tekst(20) Stanowisko; <- nazwa atrybutu
}
```

tabela:

PESEL	Nazwisko	Imię	Stanowisko
11120404045	Kowalski	Jan	lekarz
12110302021	Nowa	Andrzej	portier
33091209224	Kamieńska	Anna	lekarz
44010101102	Fajowska	Joanna	pielęgniarka

Zastosowanie struktury i utworzonych rekordów umożliwia szereg automatycznych (poprzez oprogramowanie) operacji, np.:

- znajdź rekord, dla którego wartość atrybutu „LName” jest równa „Rumiński”,
- wyświetl nazwisko i imię dla wszystkich nauczycieli z tytułem „prof.”,
- wyświetl identyfikatory nauczycieli, którzy pracują od roku 2000,
- itd.

Bazy danych – koniec wykładu 2

Co dalej?

W czasie tego wykładu przedstawiliśmy model warstwowy SZBD oraz relację pomiędzy modelem danych a strukturą danych.
Zaprezentowany został również model płaski. W czasie kolejnego wykładu poznamy model hierarchiczny i sieciowy.

ZAPRASZAMY NA WYKŁAD 3