Bazy danych i systemy zarządzania

Wykład III

Relacyjny model danych
Atrybuty, dziedziny, własności pól

Pojęcie atrybutu

Dane reprezentowane w jakiejkolwiek formie, np. w pewnym pliku, arkuszu kalkulacyjnym, bazie danych, bazie wiedzy, itp. odnoszą się zawsze do pewnego *obiektu* lub *zbioru obiektów* i charakteryzują wybrane ich *własności* będące przedmiotem zainteresowania.

Dla danego zagadnienia, zazwyczaj na etapie projektu systemu informatycznego, dokonuje się wyboru tych własności oraz stopnia szczegółowości ich reprezentacji. Do opisu własności obiektów reprezentowanych w bazie danych wykorzystuje się wybrany zestaw *atrybutów*, a dla wyrażenia ich wartości z zadaną dokładnością określa się *dziedziny atrybutów*. W praktyce często dziedzinę atrybutu definiuje się poprzez podanie *typu* danych, *rozmiaru pola*, *więzów spójności* (lokalnych, np. reguły poprawności).

Przez **atrybut obiektu** rozumie się pewną jego własność, cechę lub charakterystykę, pozwalającą częściowo opisać ten obiekt. Przykładami atrybutów mogą być kolor, kształt, waga, cena, wiek, marka, typ, numer seryjny, itp. Atrybuty przyjmują wartości z określonego zbioru (dziedziny), przy czym normalnie (choć milcząco) zakłada się, że w danym momencie dla danego obiektu wybrany atrybut przyjmuje *pojedynczą wartość*; co więcej, przyjmuje się najczęściej, że jest to *wartość atomiczna*, tzn. taka, która traktowana jest jako pojedynczy symbol i nie jest dalej analizowana po kątem jej struktury wewnętrznej, zawartości, elementów składowych, itd.

Wymaganie co do atomicznego charakteru wartości atrybutów stanowi jeden z podstawowych kanonów relacyjnych baz danych; w bardziej zaawansowanych modelach może ono być jednak osłabione, poprzez dopuszczenie wartości złożonych, takich jak zbiory, przedziały, czy nawet struktury. Wymaganie atomicznego charakter wartości atrybutów uwarunkowane jest dążeniem do maksymalnej przejrzystości modelu relacyjnych baz danych i jego zgodności z teorią, a przede wszystkim dążeniem do uzyskania dużej efektywnoći operacji algebraicznych.

Pojęcie atrybutu – zagadnienia formalne

Z matematycznego punktu widzenia atrybut jest *funkcją* a więc pewnego rodzaju odwzorowniem, które każdemu obiektowi przypisuje pewną wartość cechy którą ten atrybut definiuje.

Definicja 1 *Niech bedzie dany pewnien zbiór D_i zwany dziedziną. Attrybutem A_i o dziedzinie D_i określonym na zbiorze obiektów E nazywamy dowolną funkcję postaci:*

$$A_i \colon E \to D_i.$$

Uwaga: z definicji pojęcia *funcji* wynika, że jeżeli $A_i(e_j) = d^1_{ji}$ oraz $A_i(e_j) = d^2_{ji}$ to $d^1_{ji} = d^2_{j2}$; dla danego obiektu atrybut może przyjmować tylko jedną wartość! Wartosc ta może powtarzać się dla wielu różnych obiektów (są one wtedy *nierozróżnialne* z punktu widzenia wartości tego atrybutu).

Niech:

- $E = \{e_1, e_2, \ldots, e_m\}$ oznacza zbiór m rozważanych obiektów (encji),
- $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ oznacza zbiór n atrybutów reprezentujących wybrane własności obiektów ze zbioru E,
- $D_1, D_2, \ldots, D_n$ oznacza dziedziny powyższych atrybutów.

Zapis wartości atrybutu dla wybranego obiektu przybiera formę:

$$A_i(e_j) = d_{ji}, \tag{1}$$

Uwaga: zapis postaci:

$$A_i = d_j$$

nie mówi dla jakiego obiektu określono wartość atrybutu A_j; jest on stosowany jako *selektor* (warunek logiczny), który definiuje (wybiera) pewien podzbiór E_{ij} zbioru obiektów E, dla których ten warunek jest spełniony; matematycznie:

$$E_{ij} = \{e \in E \colon A_i(e) = d_j\}.$$

Wartości i dziedziny atrybutów

Każdy atrybut przyjmuje wartości z określonego zbioru, zadanego jawnie lub domyślnie, stanowiącego jego dziedzinę. Przyjmijmy następujące standardowe oznaczenia:

- $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ – ustalony zbiór atrybutów,
- $D_1, D_2, \ldots, D_n$ – ustalony zbiór dziedzin tych atrybutów.

Z każdym atrybutem $A_i \in \mathbf{A}$ związana jest dziedzina D_i, $i = 1, 2, \ldots, n$.

Dziedzina atrybutu może być zadana jawnie, np. poprzez wyliczenie elementów w postaci skończonego zbioru (ekstensjonalnie), lub może być zadana niejawnie w przypadku trudności wyliczenia wszystkich potencjalnych elementów tej dziedziny. Przykładem pierwszej dziedziny może być zbiór wartości atrybutu *dzień_tygodnia* w postaci {*poniedziałek, wtorek, środa, czwartek, piątek, sobota, niedziela*}. Przykładem drugiej dziedziny może być zbiór wartości atrybutu *nazwisko* – trudno tutaj wyliczyć wszystkie możliwe napisy, które mogłyby być uznane za nazwiska; z drugiej strony, przyjęcie za dziedzinę tego atrybutu zbioru napisów o ograniczonej długości maksymalnej dopuszcza napisy typu „Aaaaa" czy „xyz", które trudno uznać za realne nazwiska. W takiej sytuacji dziedzinę zadaje się zazwyczaj częściowo, poprzez określenie typu i rozmiaru pola, oraz ewentualnie regułę poprawności.

Dany atrybut dla danego obiektu może przyjmować w danej chwili czasu pojedynczą wartość – mówimy tutaj o *funkcyjnym* charakterze zależności wartości atrybutu od wybranych parametrów (obiektu, chwili czasu, ewentualnie dodatkowych czynników). Dobrymi przykładami wartości atrybutów są np. wartości zmiennych opisujących stan i zachowanie się układów fizycznych, precyzyjne dane pomiarowe, wartości wielkości fizycznych, takich jak temperatura, prędkość, napięcie, natężenie, itp. Natomiast często nie mają tego charakteru gromadzone łącznie dane pochodzące z pewnych obserwacji, np. wartości charakteryzujące *objawy* występujące łącznie u chorego jak {*gorączka, duszności, bóle, wysypka*}, czy też nieformalne opisy tekstowe.

Rodzaje dziedzin

Dziedziny nominalne i częściowo uporządkowane

Dziedzina atrybutu może tworzyć pewną strukturę; typowe z nich to:

- *dziedziny binarne* – dziedzina taka zawiera dokładnie dwa elementy, np. 0 i 1 lub określenia *tak* oraz *nie*; dodatkowo, może być narzucona relacja porządkująca te elementy, np. $0 \leq 1$,
- *dziedziny trójwartościowe* – dziedzina taka zawiera dokładnie trzy elementy, np. $\{tak,\ nie,\ brak\_danych\}$, $\{TRUE,\ FALSE,\ NULL\}$ lub $\{-, 0, +\}$; może być określona relacja porządku, np. $- \leq 0 \leq +$,
- *dziedziny nominalne (nieuporządkowane)* – są to skończone (lub rzadziej nieskończone) zbiory nazw, które nie są powiązane ze sobą żadną relacją porządkującą; typowymi przykładami mogą być np. zbiory dopuszczalnych kolorów, kształtów, itp.; zbiory takie można jednak porządkować leksykograficznie,
- *dziedziny częściowo uporządkowane* – podobnie jak wyżej, zbiory skończone lub nieskończone z relacją częściowego porządku – takie dziedziny występują stosunkowo rzadko; przykładem może być zbiór wartości „mieszanych" typu $\{list\_zwykły,\ list\_ekspresowy,\ list\_polecony,\ paczka\_zwykła,\ paczka\_wartościowa,\ przesyłka\_kurierska\}$ stanowiących wartości atrybutu *rodzaj_przesyłki* z częściowym porządkiem (względem czasu dostarczenia) ustalonym przez relację $list\_zwykły \leq list\_ekspresowy$, $list\_ekspresowy \leq przesyłka\_kurierska$, $list\_polecony \leq list\_ekspresowy$, $list\_ekspresowy \leq przesyłka\_kurierska$,
- *dziedziny tworzące strukturę kraty* – dziedziny takie wraz z wprowadzoną relacją częściowego porządku tworzą strukturę kraty; typowym przykładem mogą być specyfikacje typów,

Rodzaje dziedzin

Dziedziny uporządkowane

- *dziedziny uporządkowane* – skończone lub nieskończone zbiory wartości uporządkowanych (liniowo), np. $\{nb,nm,ns,z,ps,pm,pb\}$, gdzie nb – duży ujemny, nm – średni ujemny, ns – mały ujemny, z – około zera, ps – mały dodatni, pm – średni dodatni, pb – duży dodatni, z uporządkowaniem $np \leq nm \leq ns \leq z \leq ps \leq pm \leq pb$; może to być też pewien zbiór napisów (imion, nazwisk, nazw własnych) z relacją porządku alfabetycznego; analogiczny charakter ma zbiór ocen!
- *dziedziny liczbowe* – ze względu na liczne zastosowania i specyficzny charakter wyróżniono je jako osobny rodzaj, choć w istocie należą do klasy zbiorów uporządkowanych – wyróżnia je jednak określona arytmetyka pozwalająca realizować operacje obliczeniowe, w tym wszelkiego rodzaju przeliczenia, obliczanie wartości funkcji, obliczanie funkcji sumarycznych (takich jak: suma, średnia, wariancja, odchylenie standardowe, itp.); przykłady obejmują dowolne podzbiory zbioru liczb naturalnych, całkowitych, czy rzeczywistych, a także daty i czas, liczby zapisane w innym niż dziesiętny układzie pozycyjnym, walutę, procenty, itp. Często spotykane typy danych to:
  - *bajt*,
  - *liczba całkowita* (krótka, długa),
  - *liczba rzeczywista* (pojedynczej, podwójnej precyzji),
  - *data*,
  - *waluta*,
  - p*rocenty*

Dla dziedzin tego typu istnieje zwykle możliwość konwersji typu.

Inne typy danych

Dane *specjalizowane* nie mają charakteru atomicznego, a ich zapis i interpretacja odbywa się przy pomocy specjalnych narzędzi. Ich przetwarzanie wymaga znajomości strutury oraz specjalizowanego oprogramowania, np.:

- *dane typu tekstowego* – mogą zawierać dowolne pliki tekstowe; w bazach danych określa się je jako tzw. pola typu *memo*, przy czym nie wykonuje się na nich żadnych operacji, poza obróbką za pomocą edytora tekstu,
- *dane typu dźwięk* – zapisywane są w postaci specjalnych plików, w bazach danych jedyne ich zastosowanie polega na odtwarzaniu nagranej sekwencji; specjalistyczna obróbka możliwa jest przy zastosowaniu odpowiednich narzędzi, przy czym może ona mieć na celu np. rozpoznanie mowy i przetłumaczenie do postaci zapisu w pewnym języku symbolicznym, albo np. rozpoznanie osoby na podstawie jej wypowiedzi,
- *dane typu obraz* – zapisywane są w postaci specjalnych plików w określonym formacie, ich zastosowanie polega na odtwarzaniu (prezentacji) obrazu; specjalistyczna obróbka możliwa jest przy zastosowaniu odpowiednich narzędzi, przy czym może ona mieć na celu analizę i rozpoznawanie obrazu, porównywanie obrazów, itp. (Istnieją specjalizowane bazy danych, w których możliwe jest wyszukiwanie na podstawie wzorca),
- *dane typu sekwencja wideo* – podobie jak obrazy,
- *dane typu hyperłącze* – umożliwiają uruchomienie usługi sieciowej,
- *wiedza jako dane* – wiedza, o ile stanowi przedmiot dalszej analizy czy przetwarzania (na wyższym poziomie), może również stanowić dane wyjściowe dla tego procesu; przykładem mogą być zbiory reguł lub programy logiczne, których własności, są przedmiotem dalszej analizy. Baza danych zawierająca zbiory procedur może też być wykorzystywana do wspomagania lub automatycznej syntezy programów w oparciu o dobór istniejących komponentów o podanych specyfikacjach.

Przykładowe własności pól: ACCESS

W systemie ACCESS można zdefiniować następujące własności pól tabeli:

- **Rozmiar pola** – określa maksymalny rozmiar danych przechowywanych w polu (tekst i liczba),
- **Nowe wartości** – tylko dla *autonumer*; można wybrać generowanie przyrostowe lub losowe,
- **Format** – definiuje szablon wyświetlania zawartości danego pola,
- **Miejsca dziesiętne** – definiuje ilość miejsc po przecinku (liczby),
- **Maska wprowadzania** – definiuje szablon dla wprowadzania danych,
- **Tytuł** – pozwala podać etykietę kolumny tabeli,
- **Wartość domyślna** – pozwala zdefiniować standardową wartość wprowadzaną automatycznie,
- **Reguła poprawności** – definiuje warunek, jaki muszą spełniać wszystkie wartości w danym polu,
- **Komunikat o błędzie** – pozwala zdefiniować tekst wyświetlany przy naruszeniu reguły poprawności,
- **Wymagane** – określa czy dane pole musi zostać wypełnione, czy też może pozostać puste,
- **Zerowa długość dozwolona** – pozwala zapisać w polu łańcuch zerowej długości (zamienia puste pola tekstowe na łańcuchy zerowej długości),
- **Indeksowane** – pozwala wyspecyfikować żądanie indeksowania pola (możliwe jest indeksowanie z powtórzeniami lub bez),
- **Odnośnik** – pozwala wybrać typ formantu (dla typów tekst, liczba, logiczne) jako listę, pole wyboru, lub listę rozwijalną (pole kombi) (osiągalne przez zakładkę; może być tworzone przy pomocy kreatora).

Format pola: ACCESS

Dla pól tekstowych:

<Szablon>;<String zerowej długości>;<Null>

- *<Szablon>* – sekcja definiująca sposób wyprowadzania tekstu,
- *<String zerowej długości>* - sekcja definiująca co zostanie wyprowadzone w przypadku, gdy pole zawiera string zerowej długości (""),
- *<Null>* – sekcja definiująca co zostanie wyprowadzone w przypadku, gdy pole zawiera wartość *Null*.

Na przykład:

```
@;"Brak telefonu"[Czerwony];"Telefon nieznany"[Niebieski]
```

spowoduje wypisanie numeru telefonu zawartego w polu i odpowiednich komunikatów w przypadku jego braku lub nieznajomości.

Dla liczb obowiązuje format:

<Szablon dodatnich>;<Szablon ujemnych>;<Szablon zera>;<Null>

- *<Szablon dodatnich>* – definiuje sposób wyprowadzania liczb dodatnich,
- *<Szablon ujemnych>* – definiuje sposób wyprowadzania liczb ujemnych,
- *<Szablon zera>* – definiuje, co będzie wypisane gdy w polu jest wartość zero,
- *<Null>* – definiuje, co będzie wypisane gdy pole jest puste.

Na przykład:

```
0,00;"Ujemna cena!"[Czerwony];"Zero"[Zielony];"Brak"[Niebieski]
```

wyświetli cenę z dokładnością do dwóch miejsc po przecinku, oraz wskazane komunikaty w sytuacjach wyjątkowych.

Format danych: ACCESS

Znaczenie symboli

- `"TEKST"` – wyświetla podany w cudzysłowach tekst,
- `(spacja)` – wyświetla spację,
- `!` – wymusza lewostronne wyrównanie danych w polu,
- $*$ – wypełnia wole pole podanym znakiem,
- `\` – powoduje wyświetlenie następującego po nim znaku,
- `[kolor]` – powoduje wyświetlanie zawartości pola w zadanym kolorze (Black, Blue, Green, Cyan, Red, Magenta, Yellow, White),
- `,` – separator części dziesiętnej,
- `0` – cyfra lub zero; wystąpienie obowiązkowe,
- `#` – cyfra lub spacja,
- `%` – wyświetla liczbę w postaci procentu,
- `E-, e-, E+, e+` – notacja eksponencjalna (dla liczb),
- `@` – cały tekst, pojedynczy znak lub spacja,
- `&` – pojedynczy znak lub nic,
- `>` – powoduje wypisanie tekstu dużymi literami,
- `<` – powoduje wypisanie tekstu małymi literami,

Dla typu *data* istnieje szereg symboli specjalnych (vide: Help).

Maska wprowadzania: ACCESS

Schemat i przykłady masek

Maska wprowadzania stanowi *szablon* i *filtr* użytczny przy wprowadzaniu danych do pola; realizuje również proste *transformacje*. Maska zapewnia elementarna kontrolę poprawności wprowadzanych danych. Schemat maski wprowadzania jest następujący:

<Szablon maski>;<Przechowanie znaku>;<Znak maski>

- *<Szablon maski>* – definiuje postać maski,
- *<Przechowanie znaku>* – 0 oznacza, że znaki maski mają być przechowywane w tabeli; 1 oznacza, że znaki maski nie będą przechowywane w tabeli (standardowo 1),
- *<Znak maski>* – znak reprezentujący w masce pojedynczy symbol (standardowo _.

Na przykład, dla przechowywania numerów ISBN moż posłużyć się maską:

```
ISBN 00\-00000\-00\-0;1;\_
```

a dla numerów telefonów można posłużyć się maską:

```
(9\-99) 000\-00\-00!;0;\_
```

Maska wprowadzania może pełnić rolę (i) szablonu, (ii) filtru poprawności oraz (iii) być zastosowana do przekształcania (formatowania) wprowadzanych symboli.

Typowe maski wprowadzania dogodnie jest tworzyć przy wykorzystaniu *Kreatora masek*.

Maska wprowadzania: ACCESS

Znaczenie symboli używanych w maskach wprowadzania:

- `0` – cyfra, pozycja wymagana,
- `9` – cyfra lub spacja, pozycja nie wymagana,
- `#` – cyfra lub spacja, pozycja nie wymagana (puste miejsca konwertowane na spacje, dozwolone + i -),
- `L` – litera, pozycja wymagana,
- `?` – litera, pozycja opcjonalna,
- `A` – litera lub cyfra, pozycja wymagana,
- `a` – litera lub cyfra, pozycja opcjonalna,
- `&` – dowolny znak lub spacja, pozycja wymagana,
- `C` – dowolny znak lub spacja, pozycja opcjonalna,
- `!` – wyświetlanie znaków od prawej do lewej,
- `Password|Hasło` – powoduje utajnienie wprowadzanych znakó (są zastąpione $*$),
- `>` – powoduje, że wprowadzane za tym symbolem znaki będą konwertowane na duże,
- `<` – powoduje, że wprowadzane za tym symbolem znaki będą konwertowane na małe,
- `.,:;-/` – znak dziesiętny, separatory tysięcy, daty i czasu; zależne od ustawień w *Panelu sterowania*,
- `\` – literalne cytowanie następnego znaku.