Studia podyplomowe "Inżynieria oprogramowania" współfinansowane przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

Projekt "Studia podyplomowe z zakresu wytwarzania oprogramowania oraz zarządzania projektami w firmach informatycznych" realizowany w ramach Programu Operacyjnego Kapitał Ludzki

Konstruowanie Baz Danych

Operacje algebraiczne w bazach danych I

Antoni Ligęza
ligeza@agh.edu.pl
http://home.agh.edu.pl/~ligeza
http://home.agh.edu.pl/~ligeza/wiki

Definiowanie tablicy relacyjnej bazy danych

Definiowanie tabel

Aby zdefiniować tablicę reprezentującą relację w bazie danych należy określić następujące elementy:

1. **Schemat tabeli** postaci $R(A_1, A_2, \ldots, A_n)$, tzn.:
   - zdefiniować sekwencję atrybutów $A_1, A_2, \ldots, A_n$,
   - zdefiniować ich dziedziny $D_1, D_2, \ldots, D_n$ (określając typ, rozmiar, więzy spójności, etc. lub poprzez jawne (ekstensjonalne) zdefiniowania elementów każdego zbioru D_i, $i = 1, 2, \ldots, n$,
   - zdefiniować klucz lub indeks jednoznaczny (opcjonalnie).
2. **Krotki relacji (rekordy tabeli)**, co można uczynić:
   - *ekstensjonalnie*, np. wpisując lub wczytując ze zbioru,
   - *intensjonalnie*, definiując warunek selekcji oraz zbiór (generator) potencjalnych krotek (uniwersum), np. wczytać dane z odpowiedniego zbioru z wykorzystaniem filtru.

Predykat relacji

Predykatem relacji $R \subseteq D_1 \times D_2 \times \ldots \times D_n$ nazywamy funkcję postaci:

$$R \longrightarrow \{TRUE, FALSE\}$$

przy czym:
– $||R(d_1, d_2, \ldots, d_n)|| = TRUE$ wtw. gdy $(d_1, d_2, \ldots, d_n) \in R$, oraz
– $||R(d_1, d_2, \ldots, d_n)|| = FALSE$ wtw. gdy $(d_1, d_2, \ldots, d_n) \notin R$.
$||R(d_1, d_2, \ldots, d_n)||$ oznacza *wartość logiczną* formuły atomowej $R(d_1, d_2, \ldots, d_n)$.

©Antoni Ligęza

Suma relacji zgodnych

Operację sumy (unii) relacji można przenieść bezpośrednio z algebry zbiorów; relacje są bowiem zbiorami. Jednak ze względu na specyficzną strukturę elementów tych zbiorów oraz wynikające z nich możliwości reprezentacji w tabelach, sensownie jest zdefiniować sumę dla *relacji zgodnych*, tzn. relacji o identycznym schemacie.

Niech $R(A_1, A_2, \ldots, A_n)$ oraz $S(A_1, A_2, \ldots, A_n)$ będą dowolnymi relacjami zgodnymi, określonymi w uniwersum $U = D_1 \times D_2 \times \ldots \times D_n$, $R \subseteq U$, $S \subseteq U$. Suma relacji zgodnych definiowana jest jako suma zbiorów $R \cup S$:

$$R \cup S = \{(d_1, d_2, \ldots, d_n) \in U : (d_1, d_2, \ldots, d_n) \in R \vee (d_1, d_2, \ldots, d_n) \in S\}.$$

Sumę relacji $R \cup S$ można też zadać intensjonalnie w U poprzez zdefiniowanie predykatu relacji jak następuje:

$$||R \cup S(d_1, d_2, \ldots, d_n)|| = ||R(d_1, d_2, \ldots, d_n)|| \vee ||S(d_1, d_2, \ldots, d_n)||.$$

Przykład

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2
a_3	b_3	c_3
a_5	b_5	c_5

$\cup$

A	B	C
a_2	b_2	c_2
a_4	b_4	c_4
a_5	b_5	c_5
a_6	b_6	c_6
a_7	b_7	c_7

$=$

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2
a_3	b_3	c_3
a_4	b_4	c_4
a_5	b_5	c_5
a_6	b_6	c_6
a_7	b_7	c_7

Zgodnie z prawami algebry zbiorów, duplikaty rekordów nie są uwzględniane (standardowo). Dla poprawnej realizacji operacji liczba i typy kolumn (atrybutów) powinny być identyczne. Liczba elementów relacji wynikowej $T = R \cup S$ spełnia warunek $card(T) \leq card(R) + card(S)$.

©Antoni Ligęza

Różnica relacji zgodnych

Operację różnicy (odejmowania) relacji można przenieść bezpośrednio z algebry zbiorów (relacje są zbiorami). Jednak ze względu na specyficzną strukturę elementów tych zbiorów oraz wynikające z nich możliwości reprezentacji w tabelach, sensownie jest zdefiniować różnicę dla *relacji zgodnych*, tzn. relacji o identycznym schemacie.

Niech $R(A_1, A_2, \ldots, A_n)$ oraz $S(A_1, A_2, \ldots, A_n)$ będą dowolnymi relacjami zgodnymi, określonymi w uniwersum $U = D_1 \times D_2 \times \ldots \times D_n$, $R \subseteq U$, $S \subseteq U$. Różnica relacji zgodnych definiowana jest jako różnica zbiorów $R \setminus S$:

$$R \setminus S = \{(d_1, d_2, \ldots, d_n) \in U : (d_1, d_2, \ldots, d_n) \in R \wedge (d_1, d_2, \ldots, d_n) \notin S\}.$$

Różnicę relacji $R \setminus S$ można też zadać intensjonalnie w U poprzez zdefiniowanie predykatu relacji jak następuje:

$$||R \setminus S(d_1, d_2, \ldots, d_n)|| = ||R(d_1, d_2, \ldots, d_n)|| \wedge \neg||S(d_1, d_2, \ldots, d_n)||.$$

Przykład

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2
a_3	b_3	c_3
a_4	b_4	c_4
a_5	b_5	c_5
a_6	b_6	c_6
a_7	b_7	c_7

$\setminus$

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2
a_3	b_3	c_3
a_5	b_5	c_5
a_8	b_8	c_8
a_9	b_9	c_9

$=$

A	B	C
a_4	b_4	c_4
a_6	b_6	c_6
a_7	b_7	c_7

Dla poprawnej realizacji operacji odejmowania liczba i typy kolumn (atrybutów) powinny być identyczne. Wynikiem odejmowania $R \setminus S$ jest relacja pusta wtw. gdy $R \subseteq S$. Liczba elementów relacji wynikowej $T = R \setminus S$ spełnia warunek $card(T) \leq card(R)$.

©Antoni Ligęza

Iloczyn algebraiczny (przecięcie) relacji zgodnych

Operację iloczynu algebraicznego (zwykłego) (przecięcia) relacji można przenieść bezpośrednio z algebry zbiorów (relacje są zbiorami). Jednak ze względu na specyficzną strukturę elementów tych zbiorów oraz wynikające z nich możliwości reprezentacji w tabelach sensownie jest zdefiniować iloczyn dla *relacji zgodnych*, tzn. relacji o identycznym schemacie.

Niech $R(A_1, A_2, \ldots, A_n)$ oraz $S(A_1, A_2, \ldots, A_n)$ będą dowolnymi relacjami zgodnymi, określonymi w uniwersum $U = D_1 \times D_2 \times \ldots \times D_n$, $R \subseteq U$, $S \subseteq U$. Iloczyn relacji definiowany jest jako iloczyn zbiorów $R \cap S$, tzn.:

$$R \cap S = \{(d_1, d_2, \ldots, d_n) \in U : (d_1, d_2, \ldots, d_n) \in R \wedge (d_1, d_2, \ldots, d_n) \in S\}.$$

Iloczyn relacji $R \cap S$ można też zadać intensjonalnie w U poprzez zdefiniowanie predykatu relacji jak następuje:

$$||R(\cap S d_1, d_2, \ldots, d_n)|| = ||R(d_1, d_2, \ldots, d_n)|| \wedge ||S(d_1, d_2, \ldots, d_n)||.$$

Przykład

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2
a_3	b_3	c_3
a_4	b_4	c_4
a_5	b_5	c_5
a_6	b_6	c_6
a_7	b_7	c_7

$\cap$

A	B	C
a_1	b_1	c_1
a_3	b_3	c_3
a_5	b_5	c_5
a_6	b_6	c_6
a_8	b_8	c_8
a_9	b_9	c_9

$=$

A	B	C
a_1	b_1	c_1
a_3	b_3	c_3
a_5	b_5	c_5
a_6	b_6	c_6

Dla poprawnej realizacji operacji iloczynu liczba i typy kolumn (atrybutów) powinny być identyczne. Wynikiem operacji iloczynu jest relacja pusta wtw. gdy żaden rekord nie jest wspólny dla obu relacji. Liczba elementów relacji wynikowej $T = R \cap S$ spełnia warunek $card(T) \leq min(card(R), card(S))$. Iloczyn relacji można wyznaczyć korzystając z operacji różnicy wg wzoru: $R \cap S = R \setminus (R \setminus S)$.

©Antoni Ligęza

Dopełnienie relacji

Operację dopełnienia (uzupełnienia) relacji można przenieść bezpośrednio z algebry zbiorów (relacje są zbiorami), o ile określone jest uniwersum U, takie że $R \subseteq U$. Dopełnienie relacji R jest relacją o identycznym schemacie.

Niech $R(A_1, A_2, \ldots, A_n)$ będzie relacją określoną w uniwersum $U = D_1 \times D_2 \times \ldots \times D_n$, $R \subseteq U$. Dopełnienie relacji R definiowane jest jako relacja $\overline{R}$ będąca uzupełnieniem R do uniwersum U, tzn.:

$$R = \{(d_1, d_2, \ldots, d_n) \in U : (d_1, d_2, \ldots, d_n) \notin R\}.$$

Dopełnienie relacji R można też zadać intensjonalnie w U poprzez zdefiniowanie predykatu relacji jak następuje:

$$||\overline{R}(d_1, d_2, \ldots, d_n)|| = \neg||R(d_1, d_2, \ldots, d_n)||.$$

W praktyce, jeżeli dziedziny atrybutów $A_1, A_2, \ldots, A_n$ nie są zadane jawnie, przyjmuje się, że są one wyznaczone jako dziedziny relacji ze względu na odpowiednie argumenty, tzn. zbiory wszystkich wartości występujących w danej kolumnie tabeli.

Przykład

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$=$

A	B	C
a_1	b_1	c_2
a_1	b_2	c_1
a_1	b_2	c_2
a_2	b_1	c_2
a_2	b_2	c_1
a_2	b_1	c_1

Wynikiem operacji dopełnienia jest relacja pusta wtw. $R = U$. Liczba elementów relacji dopełnienia może być relatywnie duża i wynosi $card(U) - card(R)$, tzn. $card(D_1) \cdot card(D_2) \cdot \ldots \cdot card(D_n) - card(R)$.

©Antoni Ligęza

Iloczyn kartezjański

Operację iloczynu kartezjańskiego relacji można przenieść bezpośrednio z algebry zbiorów (relacje są zbiorami). Jednak ze względu na specyficzną strukturę elementów tych zbiorów oraz przyjęty sposób reprezentacji w tabelach wygodnie jest posłużyć się schematami relacji. Iloczyn kartezjański definiuje się dla *dowolnych* relacji (nie ma wymagania zgodności relacji).

Niech $R(A_1, A_2, \ldots, A_n)$ oraz $S(B_1, B_2, \ldots, B_m)$ będą dowolnymi relacjami, $R \subseteq U$, $S \subseteq V$. Iloczyn kartezjański relacji $T = R \times S$ definiowany jest jako relacja o schemacie $T(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$ taka, że:

$$R \times S = \{(d_1, \ldots, d_n, d_{n+1}, \ldots, d_{n+m}) : (d_1, \ldots, d_n) \in R \wedge (d_{n+1}, \ldots, d_{n+m}) \in S\}.$$

Iloczyn kartezjański relacji $R \times S$ można też zadać intensjonalnie w $U \times V$ poprzez zdefiniowanie predykatu relacji jak następuje:

$$||T(d_1, \ldots, d_n, d_{n+1}, \ldots, d_{n+m})|| = ||R(d_1, \ldots, d_n)|| \wedge ||S(d_{n+1}, \ldots, d_{n+m})||.$$

Przykład

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2
a_3	b_3	c_3

$\times$

B	C	D
b_1	c_1	d_1
b_2	c_2	d_2
b_3	c_3	d_3

$=$

A	B	C	B	C	D
a_1	b_1	c_1	b_1	c_1	d_1
a_1	b_1	c_1	b_2	c_2	d_2
a_1	b_1	c_1	b_3	c_3	d_3
a_2	b_2	c_2	b_1	c_1	d_1
a_2	b_2	c_2	b_2	c_2	d_2
a_2	b_2	c_2	b_3	c_3	d_3
a_3	b_3	c_3	b_1	c_1	d_1
a_3	b_3	c_3	b_2	c_2	d_2
a_3	b_3	c_3	b_3	c_3	d_3

Liczba elementów relacji wynikowej $T = R \times S$ spełnia warunek $card(T) = card(R) \cdot card(S)$. Uwaga: operacja iloczynu kartezjańskiego prowadzi zwykle do gwałtownego wzrostu liczby rekordów, spowolnienia operacji i produkcji powtarzającej się informacji!

©Antoni Ligęza

Projekcja

Projekcja (*rzutowanie*) jest jedną z podstawowych operacji algebry relacji. Dla intuicji, projekcja oznacza rzutowanie na wybrane kolumny, a więc ograniczenie informacji zadawanej w tabeli reprezentującej relację do wybranych kolumn. Innymi słowy, wybrane kolumny tabeli są zachowane, a inne – usuwane (pomijane; czasowo lub trwale, jeżeli wynik będzie zapisany).

Niech $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ będzie wybranym zbiorem atrybutów, a R będzie relacją o schemacie $R(A_1, A_2, \ldots, A_n)$.

Niech $\mathbf{A}' = \{A^1, A^2, \ldots, A^k\}$ będzie zbiorem atrybutów wybranych ze zbioru $\mathbf{A}$, $\mathbf{A}' \subseteq \mathbf{A}$ (ograniczona liczba atrybutów i zmieniona ew. kolejność).

Niech P będzie relacją o schemacie $P(A^1, A^2, \ldots, A^k)$.

Operację projekcji π_P definiujemy jako operację postaci $\pi_P : \; R \longrightarrow P$ przekształcającą relację R w pewną relację P, taką, że:

$$\pi_P((d_1, d_2, \ldots, d_n)) = (d^1, d^2, \ldots, d^k),$$

gdzie, że $d^i = d_j$ dla $A^i = A_j$, $i = 1, 2, \ldots, k$, $j \in \{1, 2, \ldots, n\}$; $(d_1, d_2, \ldots, d_n) \in R$, $(d^1, d^2, \ldots, d^k) \in P$.

Oznaczenia:

Dla reprezentacji operacji projekcji relacji R zdefiniowanej jak wyżej stosowane są następujące oznaczenia:

- $\pi_{A^1, A^2, \ldots, A^k}(R)$ – rzut relacji R na atrybuty $A^1, A^2, \ldots, A^k$,
- $\pi_{i_1, i_2, \ldots, i_k}(R)$ – rzut relacji R na składowe $i_1, i_2, \ldots, i_k$,
- $R[A^1, A^2, \ldots, A^k]$ – rzut relacji R na atrybuty $A^1, A^2, \ldots, A^k$.

  Przykładowo, niech R będzie relacją o schemacie $R(A, B, C, D, E)$; wówczas

  $$\pi_{3,5,1}(R) = \pi_{C,E,A}(R) = R[C, E, A]$$

  Logicznie $||R(d_1, d_2, \ldots, d_n)[A^1, A^2, \ldots, A^k]||$ $(||R(d^1, d^2, \ldots, d^k)||)$ wtw. $\exists x_1, x_2, \ldots, x_{n-k} R(x_1, \ldots, x_{i_1}, d^1, x_{i_1+1}, \ldots, x_{i_2}, d^2, \ldots, x_{i_k}, d^k, \ldots, x_{n-k})$.

©Antoni Ligęza

Projekcja: przykład, własności, problemy

Przykład: Niech R będzie relacją o schemacie $R(A, B, C, D, E, F)$:

A	B	C	D	E	F
a_1	b_1	c_1	d_1	e_1	f_1
a_1	b_1	c_1	d_2	e_2	f_2
a_1	b_1	c_1	d_3	e_3	f_3
a_2	b_2	c_2	d_1	e_1	f_1
a_2	b_2	c_2	d_2	e_2	f_2

Projekcją (rzutem) relacji R na atrybuty E, C, A jest relacja $\pi_{E,C,A}(R)$ ($\pi_{5,3,1}(R)$, $R[E, C, A]$) równa:

E	C	A
e_1	c_1	a_1
e_2	c_1	a_1
e_3	c_1	a_1
e_1	c_2	a_2
e_2	c_2	a_2

Własności operacji projekcji:

- zmienia schemat relacji (ogranicza liczbę atrybutów i ich kolejność,
- tracona jest informacja o wartościach niektórych atrybutów,
- mogą powstać duplikaty (rekordy nierozróżnialne),
- liczba rekordów nie zmienia się (o ile nie usuniemy duplikatów (opcja `DISTINCT` w klauzuli `SELECT`; wówczas liczba rekordów maleje).

Operacja projekcji może prowadzić do utraty możliwości rozróżniania niektórych rekordów; może ona stanowić pewien operator *abstrakcji* (stosowany celowo); ale może też prowadzić do niezamierzonej utraty informacji.

©Antoni Ligęza

Antyprojekcja

Antyprojekcja jest stosunkowo rzadko spotykaną operacją algebry relacji. Niech R będzie relacją o schemacie $R(A_1, A_2, \ldots, A_n, B)$, gdzie B jest pewnym wyróżnionym atrybutem (może on występować na dowolnej pozycji).

Niech $D_1, D_2, \ldots, D_n$ będą dziedzinami kolejnych atrybutów $A_1, A_2, \ldots, A_n$, tzn.:

$$D_i = \pi_i(R),$$

oraz niech $U = D_1 \times D_2 \times \ldots \times D_n$ będzie uniwersum względem atrybutów $A_1, A_2, \ldots, A_n$.

Niech $D_B = \pi_B(R)$ będzie zbiorem wszystkich wartości atrybutu B występujących w relacji R.

Operacja antyprojekcji relacji R której wynikiem jest relacja $R]B[$ (antyprojekcja R na atrybut B) jest zdefiniowana jako:

$$R]B[= \{b \in B : \ \forall (d_1, d_2, \ldots, d_n) \in U, \ (d_1, d_2, \ldots, d_n, b) \in R\}.$$

Przykład: Niech $R(A_1, A_2, B)$ będzie relacją zdefiniowaną jak następuje:

A_1	A_2	B
a_1	a_1	b_1
a_2	a_1	b_1
a_1	a_2	b_1
a_2	a_2	b_1
a_1	a_1	b_2
a_2	a_1	b_2
a_1	a_2	b_2

Wynikiem operacji antyprojekcji R na B jest relacja $R]B[$ jak poniżej:

B
b_1

Logicznie $||R]B[(b)||$ wtw. gdy $\forall (d_1, d_2, \ldots, d_n) \in U, \ (d_1, d_2, \ldots, d_n, b) \in R$.

©Antoni Ligęza

Interpretacja operacji antyprojekcji

Operacja antyprojekcji pozwala odpowiadać na pytania typu:

> *Którzy dostawcy dostarczają* **wszystkie** *produkty (występujące w tabeli)?*

Przykład: Niech będzie dana tabela dostawców wraz ze specyfikacją dostarczanego produktu:

Dostawca	*Element*
Budex	*cegła*
Budex	*pustak*
Budex	*cement*
Budex	*piasek*
Matbud	*cegła*
Matbud	*pustak*
Matbud	*cement*
Matbud	*gips*
Matbud	*piasek*
Matbud	*żwir*
Probud	*cegła*
Probud	*cement*
Probud	*piasek*
Probud	*żwir*

Wynikiem antyprojekcji tej tabeli na atrybut *Dostawca* jest tabela:

Dostawca
Matbud

Antyprojekcja pozwala odpowiadać na pytania zawierające kwantyfikator ogólny ($\forall$).

©Antoni Ligęza

Iloraz

Operacja ilorazu umożliwa *dzielenie* relacji przez relację.

Niech R będzie relacją o schemacie $R(A_1, A_2, \ldots, A_n)$, a S relacją o schemacie $S(B_1, B_2, \ldots, B_k)$, przy czym zakładamy że $n > k$ oraz $S \neq \emptyset$.

Wynikiem operacji dzielenia relacji R przez relację S jest relacja $R \div S$ będąca zbiorem wszystkich $(n-k)$-krotek t, takich, że dla **wszystkich** k-krotek $s \in S$, krotka $ts \in R$.

Niech $\pi_{1,2,\ldots,n-k}(R) = T$.
T jest zbiorem wszystkich początkowych ciągów krotek relacji R (o długości $n-k$). Wówczas relacja

$$(T \times S) \setminus R$$

jest zbiorem wszystkich n-krotek nie należących do R, utworzonych poprzez konkatenację pierwszych $n-k$ składowych z relacji R z **wszystkimi** krotkami z S.

Niech

$$V = \pi_{1,2,\ldots,n-k}((T \times S) \setminus R);$$

V jest zbiorem wszystkich $n-k$-krotek v, takich, że dla pewnej krotki $s \in S$, konkatenacja $vs \notin R$. Zatem

$$T \setminus V = R \div S,$$

lub:

$$R \div S = \pi_{1,2,\ldots,n-k}(R) \ \setminus \ \pi_{1,2,\ldots,n-k}((\pi_{1,2,\ldots,n-k}(R) \times S) \setminus R)$$

Operacja dzielenia relacji R przez relację S służy do znajdowania wszystkich początkowych ciągów z relacji R o długości $n-k$ (wszystkich fragmentów lub początków), które w R są skonkatenowane ze **wszystkimi** krotkami **zadanymi w relacj**i S.

©Antoni Ligęza

Iloraz: przykład i interpretacja

Operacja dzielenia relacji R przez relację S ma istotne znaczenie praktyczne i stanowi rozszerzenie operacji antyprojekcji. W przypadku operacji antyprojekcji, można było uzyskać odpowiedź na pytanie typu:

Którzy dostawcy dostarczają **wszystkie** *produkty?*

Natomiast w przypadku operacji ilorazu można uzyskać odpowiedź na pytanie typu:

Którzy dostawcy dostarczają produkty z **określonego zbioru***?*

Interesujący nas zbiór zadawany jest właśnie relacją S. Poniżej przedstawiono prosty przykłd.

Przykład:

Dostawca	*Element*
Budex	*cegła*
Budex	*pustak*
Budex	*cement*
Budex	*piasek*
Matbud	*pustak*
Matbud	*cement*
Matbud	*gips*
Matbud	*piasek*
Probud	*cegła*
Probud	*cement*
Probud	*piasek*
Probud	*żwir*

$\div$

Element
cegła
cement
piasek

$=$

Dostawcya
Budex
Probud

Operacja dzielenia pozwala zatem na realizację zapytań formalizowalnych logicznie z zastosowaniem kwantyfikatora ogólnego ($\forall$).

©Antoni Ligęza

Uogólniona suma relacji

Zdefiniujemy uogólnioną sumę relacji (niekoniecznie zgodnych), czyli relacji o różnych schematach. Niech $R(A_1, A_2, \ldots, A_n)$ oraz $S(B_1, B_2, \ldots, B_m)$ będą dowolnymi relacjami, $\mathbf{A} = \{A_1, A_2, \ldots, A_n\}$ oraz $\mathbf{B} = \{B_1, B_2, \ldots, B_m\}$. Niech $\mathbf{B'} = \mathbf{B} \setminus \mathbf{A}$, tzn. zbiór $\mathbf{B'}$ zawiera wszystkie atrybuty zbioru $\mathbf{B}$ nie występujące w $\mathbf{A}$. Niech $\mathbf{B'} = \{B^1, B^2, \ldots, B^k\}$, $k \leq m$.

Niech $R + S$ będzie nową relacją o schemacie zdefiniowanym przez $(R + S)(A_1, A_2, \ldots, A_n, B^1, B^2, \ldots, B^k)$. Relacja $R + S$ będzie nazywana *uogólnioną sumą* relacji R i S, jeżeli dla każdego rekordu $rs \in R + S$ spełniony jest warunek:

$$\pi_{A_1,A_2,\ldots,A_n}(rs) \in R$$

lub

$$\pi_{B_1,B_2,\ldots,B_m}(rs) \in S$$

Przykład:

A	B
a_1	b_1
a_2	b_2
a_3	b_3

$+$

B	C
b_1	c_1
b_2	c_2
b_3	c_3

$=$

A	B	C
a_1	b_1	c_1
a_1	b_1	c_2
a_1	b_1	c_3
a_1	b_2	c_2
a_1	b_3	c_3
a_2	b_1	c_1
a_2	b_2	c_1
a_2	b_2	c_2
a_2	b_2	c_3
a_2	b_3	c_3
a_3	b_1	c_1
a_3	b_2	c_2
a_3	b_3	c_1
a_3	b_3	c_2
a_3	b_3	c_3

©Antoni Ligęza

Interpretacja sumy uogólnionej: przykład

Uogólniona suma relacji jest relacją o schemacie zawierającym wszystkie atrybuty obu relacji (powtarzające się atrybuty występują tylko raz) oraz wszystkie krotki zbudowane z elementów relacji R i S, takie, że obcięcie danej krotki do zbioru atrybutów relacji R należy do tej relacji lub obcięcie do zbioru atrybutów relacji S należy do tej relacji. Uogólniona suma relacji odpowiada na pytanie typu:

> *Jakich odbiorców mogłyby (potencjalnie) znaleźć produkty dostarczane przez pewnych dostawców oraz którzy dostawcy mogliby (potencjalnie) dotarczać produkty dla pewnych odbiorców?*

Przykład:

Dostawca	*Produkt*
Budex	*cegła*
Budex	*pustak*
Budex	*cement*
Budex	*piasek*
Matbud	*pustak*
Matbud	*cement*
Matbud	*gips*
Matbud	*piasek*

+

Produkt	*Odbiorca*
cegła	*Wykbud*
pustak	*Wykbud*
cement	*Rembud*
cement	*Wykbud*
piasek	*Rembud*
piasek	*Wykbud*
gips	*Rembud*

=

Dostawca	*Produkt*	*Odbiorca*
Budex	*cegła*	*Rembud*
Budex	*pustak*	*Rembud*
Budex	*cement*	*Rembud*
Budex	*piasek*	*Rembud*
Matbud	*pustak*	*Rembud*
Matbud	*cement*	*Rembud*
Matbud	*gips*	*Rembud*
Budex	*gips*	*Rembud*
Matbud	*piasek*	*Rembud*
Budex	*cegła*	*Wykbud*
Budex	*pustak*	*Wykbud*
Budex	*cement*	*Wykbud*
Budex	*piasek*	*Wykbud*
Matbud	*pustak*	*Wykbud*
Matbud	*cement*	*Wykbud*
Matbud	*gips*	*Wykbud*
Matbud	*piasek*	*Wykbud*

©Antoni Ligęza

Wyszukiwanie danych: operacja selekcji

Operacja **selekcji** jest podstawową operacją realizowaną w relacyjnych bazach danych. Jest ona stosowana do wyszukiwania danych spełniających zadane *kryteria* (warunki). Operację selekcji można więc traktować jako sposób *inetnsjonalnego* definiowania danych (interesującej nas relacji) w uniwersum zadawanym *ekstensjonalnie* przez relację wyjściową.

Niech R będzie relacją o schemacie $R(A_1, A_2, \ldots, A_n)$. Zauważmy, że atrybuty $A_1, A_2, \ldots, A_n$ można traktować jak *zmienne*, które dla poszczególnych rekordów przybierają określone wartości atomiczne. Niech Ψ oznacza pewną formułę logiczną zbudowaną ze:

- stałych (w tym liczb i łańcuchów znakowych),
- atrybutów $A_1, A_2, \ldots, A_n$ pełniących rolę zmiennych; w formule Φ są one zmiennymi wolnymi (stosowana notacja: $[A_i]$),
- funkcji wbudowanych w systemie (określonych na stałych, zmiennych oraz wartościach funkcji),
- symboli relacji wbudowanych w systemie (np. $<$, $>=$, etc.),
- spójników logicznych dopuszczalnych w systemie, oraz
- nawiasów definiujących kolejność operacji.

Formuła Ψ definiuje **kryterium selekcji**. Poprawnie zdefiniowana formuła Ψ definiuje jednoznacznie relację R_Ψ, taką, że:

$$R_\Phi = \{(d_1, d_2, \ldots, d_n) \in R : \Phi\{A_1/d_1, A_2/d_2, \ldots, A_n/d_n\}\}. \qquad (1)$$

Zapis $\{A_1/d_1, A_2/d_2, \ldots, A_n/d_n\}$ oznacza podstawienie stałych $d_1, d_2, \ldots, d_n$ za zmienne $A_1, A_2, \ldots, A_n$. Operacja selekcji definiowana jest poprzez zadanie formuły Ψ a jej wynik poprzez równość (1).

©Antoni Ligęza

Operacja selekcji: definiowanie i własności

Z fomułą Ψ można jednoznacznie związać *funkcję selekcji* σ_Ψ, określoną jak następuje:

$$\sigma_\Psi : \ R \longrightarrow R_\Psi.$$

Funkcja selekcji (wyboru) może być także interpretowana jako funkcja charakterystyczna poszukiwanej relacji, tzn.:

$$\sigma_\Psi(d_1, d_2, \ldots, d_n) = \begin{cases} 1 & \text{jeżeli } (d_1, d_2, \ldots, d_n) \in R_\Psi \\ 0 & \text{jeżeli } (d_1, d_2, \ldots, d_n) \notin R_\Psi \end{cases}$$

Tak więc przy zadanej wyjściowej relacji R oraz warunku selekcji Ψ, operacja selekcji może być zapisana w postaci:

$$\sigma_\Psi(R) = R_\Psi,$$

gdzie R_Ψ jest zbiorem tych wszystkich n-krotek relacji R, które spełniają warunek Ψ, $R_\Psi \subseteq R$.

W postaci logicznej operację selekcji można też zdefiniować jako:

$$||\sigma_\Psi(R))|| = ||R|| \wedge ||\Psi||.$$

Własności operacji selekcji:

- zostaje zachowany schemat relacji, oraz
- w odniesieniu do wyselekcjonowanych krotek nie jest tracona informacja (o ile jednocześnie nie jest zastosowana operacja projekcji, zostaje zachowany ich pełny opis),
- zazwyczaj znacznemu zmniejszeniu ulega liczba rekordów (w odniesieniu do relacji wyjściowej),
- wynikiem operacji selekcji może być zbiór zawierający 0, 1, lub wiele krotek relacji wyjściowej.

©Antoni Ligęza

Przykład operacji selekcji

Rozważmy relację R zadaną tablicą `Pracownik`:

ID_prac	Nazwisko	Imię	Data ur	Stanowisko	Dział	Stawka
MT101	Abacki	Adam	61-01-01	robotnik	P10	550,00 zł
MT102	Abakowski	Alojzy	61-01-02	robotnik	P10	574, 00 zł
MT103	Adamski	Antoni	61-01-03	robotnik	P20	1275,00 zł
MT104	Adamski	Arnold	61-01-03	robotnik	P20	1280,00 zł
MT105	Adamski	Arnold	61-01-03	robotnik	P20	1295,00 zł
KT101	Aron	Antonina	61-01-03	robotnik	P10	575,00 zł
MU101	Batman	Bogusław	67-02-13	kierownik	P30	1224,00 zł
KU101	Celińska	Mirosława	69-03-08	analityk	F10	975,00 zł
MV101	Dioniziak	Dariusz	71-10-17	v-prezes	V	3000,00 zł

Niech Ψ_1 = *Nazwisko='Adamski'*; wówczas $\sigma_1(R)$ jest tablicą postaci:

ID_prac	Nazwisko	Imię	Data ur	Stanowisko	Dział	Stawka
MT103	Adamski	Antoni	61-01-03	robotnik	P20	1275,00 zł
MT104	Adamski	Arnold	61-01-03	robotnik	P20	1280,00 zł
MT105	Adamski	Arnold	61-01-03	robotnik	P20	1295,00 zł

Podobnie, Ψ_2 =*(Stanowisko='robotnik' AND Stawka<1000,00 zł) OR (Stanowisko='analityk' AND Stawka<1000,00 zł)* da wynik $\sigma_2(R)$:

ID_prac	Nazwisko	Imię	Data ur	Stanowisko	Dział	Stawka
MT101	Abacki	Adam	61-01-01	robotnik	P10	550,00 zł
MT102	Abakowski	Alojzy	61-01-02	robotnik	P10	574, 00 zł
KT101	Aron	Antonina	61-01-03	robotnik	P10	575,00 zł
KU101	Celińska	Mirosława	69-03-08	analityk	F10	975,00 zł

©Antoni Ligęza

Definiowanie warunku selekcji

Z logicznego punktu widzenia, warunek selekcji Ψ musi być *poprawnie zdefiniowaną formułą*, której wartość logiczna jest określona dla wszystkich rekordów tabeli wyjściowej. W praktyce, definiowanie kryterium selekcji odbywa się z wykorzystaniem:

- predefiniowanego szablonu użytkownika,
- właściwego dla systemu szablonu *Query by Example*,
- języka SQL (instrukcji wprowadzonej ręcznie, generowanej automatycznie lub z poziomu języka wyższego zrędu),
- specyficznego dla systemu języka zapytań (np. VTLS, Datalog, Prolog).

Typowe elementy definiowania warunku selekcji obejmują:

- użycie stałych:
  - numerycznych, np. 13, +13, -13, 13.57 (13,57),
  - tekstowych, np. 'Kowalski', "Kowalski",
  - logicznych, *True*, *False* (-1, 0) lub (0,1),
  - datowych, np. #99-08-13#, '1999-08-13', '1999/08/13',
- nazw atrybutów i wyrażeń ścieżkowych, np. `Nazwisko`, `[Data ur]`, `Pracownik.Nazwisko`,
- operatorów:
  - arytmetycznych, np. +, -, *, /,
  - tekstowych, np. +, &, %, _
  - logicznych, np. *and*, *or*, *not*, *xor*,
  - relacyjnych, np.=, <, >, <=, >=, <>, !=,
  - specjalnych (porównania), np. *Like*, *Between*,
- funkcji, np. *Abs(.)*, *Sqr(.)*, *Trim(.)*, itp.

©Antoni Ligęza

Idea szablonu QBE

Definiowanie szablonu QBE

Aby zdefiniować szablon zapytania używając języka QBE należy określić:

- **Atrybuty** – zdefiniować, które pola mają być uwzględnione w definiowanym zapytaniu; podaje się też ich kolejność,
- **Źródło atrybutu** – tabela lub zapytanie z którego pochodzi dany atrybut (uwaga: w zapytaniu mogą występować atrybuty o takich samych nazwach, ale pochodzące z różnych zródeł),
- **Sortowanie** – czy i jak wyniki zapytania mają być posortowane,
- **Wyświetlanie** – czy dane pole ma być wyświetlane,
- **Kryterium selekcji** – formuły definiujące warunek selekcji, a więc *intensjonalną* definicję relacji docelowej.

Schemat szablonu QBE

$Pole:$	A_1	A_2	$\ldots$	A_j	$\ldots$	A_k
$Tabela:$	T_1	T_2	$\ldots$	T_j	$\ldots$	T_k
$Sortuj:$			$\ldots$	$r/m/b$	$\ldots$	
$Pokaż:$			$\ldots$	y/n	$\ldots$	
$Kryteria:$	k_1	k_2	$\ldots$	k_j	$\ldots$	k_k
$Lub:$	k'_1	k'_2	$\ldots$	k'_j	$\ldots$	k'_k

(2)

W wyniku działania kwerendy otrzymuje się tabelę o schemacie $K(A_1, A_2, \ldots, A_k)$; w zależności od tego, ile rekordów spełnia określone kryteria, w tabeli wynikowej może być 0, 1, lub wiele rekordów. Mogą one podlegać dalszemu przetwarzaniu.

©Antoni Ligęza

System ACCESS – użycie wyrażeń

Definiowanie wyrażeń

Wyrażenia to definicje funkcji określające sposób wyliczania pewnych wartości; mogą one być konstruowane za pomocą *stałych*, atrybutów (pełniących rolę zmiennych), operatorów (arytmetycznych, logicznych, etc.) oraz predefiniowanych (w systemie) funkcji.

Stałe obejmują:

- **stałe numeryczne**, np. 13, +13, −13, +13, 34, 13.35,
- **stałe tekstowe**, np. "*Kowalski*", "Kraków", "*jan kowalski*",
- **stałe logiczne** – PRAWDA/True *oraz* FAŁSZ/False*; liczbowymi odpowiednikami są* −1 *dla prawdy oraz* 0 *dla fałszu,*
- **stałe datowe** – *określają wartość daty i występują w ogranicznikach* #*, np.* #07-22-98#*.*

Operatory obejmują:

- **operatory arytmetyczne** – dodawanie (+), odejmowanie (−), mnożenie (∗), dzielenie (/), potęgowanie (^),
- **operatory relacyjne** – =, >, <, <=, >=, <>, *BETWEEN ... AND ...*, *IS NULL*, *IS NOT NULL*, ich wynik jest wartością logiczną (PRAWDA lub FAŁSZ); mogą być stosowane do porównywania liczb (dat, walut) i tekstów,
- **operatory tekstowe** – +, &, są to operatory konkatenacji tekstów,
- **operatory datowe** – odejmowanie dat (−) (wynik w dniach), modyfikacja daty (+, −), wynikiem jest data,
- **operatory logiczne** – koniunkcja (*AND*), alternatywa (*OR*) i negacja (*NOT*).

©Antoni Ligęza

System ACCESS – typy kwerend

Typy kwerend w systemie ACCESS

- **Kwerenda wybierająca**. Kwerenda taka zwraca dane z jednej lub wielu tabel. Realizuje operacje projekcji i selekcji; dodatkowo może dostarczać pól wyliczeniowych. Pozwala uzyskać efekt *perspektywy*; dane zmodyfikowane w tabeli będącej wynikiem zapytania zostaną zmodyfikowane (bez ostrzeżenia!) w tabelach wyjściowych. Pozwala również na agregację danych.
- **Kwerenda krzyżowa** – specjalny typ kwerendy wyszukującej i jednocześnie grupującej dane wg specyficznych wartości z wybranej kolumny; wartości te stają się nagłówkami kolumn (transpozycja),
- **Kwerendy funkcjonalne**. Realizują różne operacje prowadzące do *trwałych zmian* w tabelach.
  - **Kwerenda tworząca tabelę** (na podstawie innej tabeli),
  - **Kwerenda usuwająca** – usuwa dane z tabeli,
  - **Kwerenda dołączająca** - dołącza dane z tabeli do innej tabeli,
  - **Kwereneda aktualizująca** – modyfikuje dane w tabeli.
- **Kwerendy SQL**
  - **Kwerenda składająca** – realizuje sumę tabel,
  - **Kwerenda przekazująca** – przekazuje instrukcję SQL do serwera zewnętrznego,
  - **Kwerenda definiująca dane** – pozwala definiować tabele.
- **Kwerenda parametryczna** – pozwala tworzyć kwerendy dynamicznie aktualizujące parametry zapytania z wykorzystaniem okna dialogowego (argument definiowany jest w postaci `[zapytanie o argument]`.

©Antoni Ligęza

System ACCESS – funkcje wbudowane

Funkcje arytmetyczne

- `Exp(n)` – oblicza e podniesione do potęgi n,
- `Log(n)` – oblicza logarytm naturalny z n,
- `Sqr(n)` – oblicza pierwiastek kwadratowy z n,
- `Int(n)` – oblicza część całkowitą argumentu; $Int(n) \leq n$,
- `Fix(n)` – oblicza część całkowitą argumentu; $Fix(n) \leq |n|$,
- `Abs(n)` – oblicza wartość bezwzględną z n,
- `Sgn(n)` – zwraca znak liczby ($+1$, -1 lub 0).

Funkcje konwersji

- `Chr(n)` – zamienia wartość n na odpowiadający znak ASCII,
- `Asc(c)` – zwraca kod ASCII znaku c,
- `Str(n)` – zamienia liczbę n na łańcuch znaków,
- `Val(c)` – zamienia łańcuch znaków (zgodny z formatem liczby!) na tą liczbę,
- `Clng(w)` – zamienia argument na liczbę całkowitą długą,
- `DateSerial(rok; miesiąc; dzień)` – wyznacza datę dla podanych argumentów,
- `Day(n), Month(n), Weekday(n), Year(n)` – wyznacza element daty odpowiadający liczbie n.

©Antoni Ligęza

System ACCESS – funkcje wbudowane

Funkcje operujące na łańcuchach znaków

- `Space(n)` – zwraca ciąg spacji o długości n,
- `String(n; c)` – zwraca ciąg n powtórzeń pierwszego znaku c,
- `InStr(n; c1; c2; jak)` – wyznacza pozycje wystąpienia podłańcucha $c2$ w $c1$ począwszy od znaku n; parametr jak ustala, czy uwzględniania jest wielkość liter (jak=0,1,2),
- `Left(c; n)` – zwraca n pierwszych od lewej znaków łańcucha c,
- `Right(c; n)` – zwraca n znaków łańcucha c od prawej,
- `Mid(c; n1; n2)` – zwraca $n2$ znaków łańcucha c poczawszy od znaku $n1$,
- `LTrim(c), RTrim(c), Trim(c)` – usuwa spacje wiodące, spacje z prawej strony lub obustronne,
- `LCase(c)` – zamienia litery łańcucha c na małe,
- `UCase(c)` – zamienia litery łańcucha c na duże,
- `Len(c)` – zwraca długość łańcucha c,
- `StrComp(c1; c2; jak)` – zwraca -1 lub 1 w zależności od tego czy łańcuch $c1$ poprzedza $c2$ czy odwrotnie; zwraca 0 dla identycznych łańcuchów. Parametr jak = 0,1,2 definiuje sposób porównywania (0 – uwzględnia wielkość liter, 1 – nie uwzględnia wielkości liter, 2 – zgodnie z ustawieniami systemowymi bazy danych).

©Antoni Ligęza

System ACCESS – funkcje wbudowane

Funkcje daty i czasu

- `Date(), Time(), Now()` – zwracają dane z zegara systemowego (datę, czas, datę i czas),

Funkcje sprawdzające

- `IsDate(arg), IsEmpty(arg), IsNull(arg), IsNumric(arg)` – sprawdzają, czy podany argument jest odpowiednio: datą, łańcuchem pustym, ma wartość *Null*, oraz czy jest liczbą.

Funkcja wyboru

- `Iif(l; w1; w2)` – jeżeli wyrażenie logiczne l jest prawdziwe, zwraca $w1$; w przeciwnym przypadku zwraca $w2$. Przykład zastosowania: `Iif(IsNull([Cena]); "Brak ceny"; Str([Cena])+"PLN")`.

Operator porównania stringów Like

- `Like "wzorzec"` – porównuje poprzedzający argumen ze wzorcem; we wzorcu można użyć symboli wieloznacznych,
- `*` – dowolny ciąg znaków, również pusty,
- `?` – dokładnie jeden znak,
- `[]` – dowolny ze znaków w nawiasach ([xyz] – x, y albo z),
- `[!]` – żaden ze znaków po "!" ([!xyz] – nie x, ani y ani z),
- `[a-z]` – znak z zakresu od a do z ([0-9] – dowolna cyfra),
- `#` – dowolna cyfra.

©Antoni Ligęza

Studia podyplomowe "Inżynieria oprogramowania" współfinansowane przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

Projekt "Studia podyplomowe z zakresu wytwarzania oprogramowania oraz zarządzania projektami w firmach informatycznych" realizowany w ramach Programu Operacyjnego Kapitał Ludzki

©Antoni Ligęza