ĆWICZENIE NR 10

JĘZYKI KWEREND

Iwona Pająk

1. CEL ĆWICZENIA

Celem ćwiczenia jest przedstawienie podstaw teoretycznych budowy relacyjnych języków manipulowania danymi a w szczególności języków zapytań. Omówione zostały dwa popularne języki: SQL oraz QBE. Dodatkowo zamieszczone zostały przykłady zapytań wyrażonych w omówionych wcześniej językach.

2. WPROWADZENIE

Systemy zarządzania bazami danych oprócz możliwości definiowania schematów danych posiadają również wbudowane mechanizmy manipulowania tymi danymi. Wyszukiwanie danych spełniających określone warunki, wprowadzanie, usuwanie oraz aktualizowanie informacji może być zrealizowane w bazach dzięki *językom manipulowania danymi* (DML – Data Manipulation Language). Przeciętny użytkownik najczęściej jednak korzysta z mechanizmów wyszukiwania. Mechanizmy te są realizowane poprzez tzw. *języki zapytań* (query language). Konkretne pytanie kierowane przez użytkownika do bazy jest nazywane *zapytaniem* lub *kwerendą*.

W celu przedstawienia minimalnych możliwości jakie powinien posiadać język zapytań Codd [5] zaproponował trzy abstrakcyjne języki, dla których przedstawił następującą klasyfikację:

1. język algebraiczny (zapytania wyrażane są w postaci wyrażeń, w których występują relacje i specjalne operatory relacyjne),

2. języki oparte na rachunku predykatów (zapytania opisują żądany zbiór krotek przez podanie predykatu (warunku), który krotki muszą spełniać):
   2.1. język oparty na relacyjnym rachunku krotek (obiektami operacji są krotki relacji)
   2.2. język oparty na relacyjnym rachunku dziedzin (obiektami operacji są elementy dziedzin atrybutów).

Wspólną cechą wymienionych języków jest to, że gwarantują one znalezienie dowolnej informacji o ile znajduje się ona w bazie (tzw. pełność). Rzeczywiste języki zapytań oparte są na językach abstrakcyjnych (co gwarantuje, że są pełne) ale posiadają zwykle jakieś dodatkowe rozszerzenia.

2.1 JĘZYKI ALGEBRAICZNE

Języki algebraiczne wykorzystują tzw. operatory relacyjne do budowy zapytań. Podstawowymi operatorami są dwa operatory jednoargumentowe: rzutowania i selekcji oraz jeden dwuargumentowy operator złączenia. Dodatkowo zestaw operatorów jest rozszerzany operatorami teoriomnogościowymi.

W celu zdefiniowania operatorów relacyjnych używane będą następujące oznaczenia:

$\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$	relacja $\mathbf{R}$ określona na zbiorze atrybutów $A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}$,
$\mathbf{S}\left(A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}\right)$	relacja $\mathbf{S}$ określona na zbiorze atrybutów $A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}$,
$k^{\mathbf{R}}$, $k^{\mathbf{S}}$	krotka (wiersz) relacji odpowiednio $\mathbf{R}$ i $\mathbf{S}$, $k^{\mathbf{R}} = \left\langle a_1^{\mathbf{R}}, a_2^{\mathbf{R}}, \ldots, a_n^{\mathbf{R}} \right\rangle$, $k^{\mathbf{S}} = \left\langle a_1^{\mathbf{S}}, a_2^{\mathbf{S}}, \ldots, a_m^{\mathbf{S}} \right\rangle$, $a_1^{\mathbf{R}} \in A_1^{\mathbf{R}}$, $a_2^{\mathbf{R}} \in A_2^{\mathbf{R}}$, …, $a_n^{\mathbf{R}} \in A_n^{\mathbf{R}}$, $a_1^{\mathbf{S}} \in A_1^{\mathbf{S}}$, $a_2^{\mathbf{S}} \in A_2^{\mathbf{S}}$, …, $a_m^{\mathbf{S}} \in A_m^{\mathbf{S}}$,
$\tilde{A}^{\mathbf{R}}$	podzbiór zbioru atrybutów $A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}$, $\tilde{A}^{\mathbf{R}} = A_i^{\mathbf{R}}, \ldots, A_j^{\mathbf{R}}$,

$\tilde{A}^{\mathbf{S}}$	podzbiór zbioru atrybutów: $A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}$, $\tilde{A}^{\mathbf{S}} = A_k^{\mathbf{S}}, \ldots, A_l^{\mathbf{S}}$,
$k^{\mathbf{R}}[\tilde{A}^{\mathbf{R}}]$	ograniczenie krotki $k^{\mathbf{R}} \in \mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$ do zbioru atrybutów $\tilde{A}^{\mathbf{R}}$ tzn.: $k^{\mathbf{R}}[\tilde{A}^{\mathbf{R}}] = \left\langle a_i^{\mathbf{R}}, \ldots, a_j^{\mathbf{R}} \right\rangle$, gdzie $a_i^{\mathbf{R}} \in A_i^{\mathbf{R}}$, …, $a_j^{\mathbf{R}} \in A_j^{\mathbf{R}}$
$k^{\mathbf{S}}[\tilde{A}^{\mathbf{S}}]$	ograniczenie krotki $k^{\mathbf{S}} \in \mathbf{R}\left(A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}\right)$ do zbioru atrybutów $\tilde{A}^{\mathbf{S}}$ tzn.: $k^{\mathbf{S}}[\tilde{A}^{\mathbf{S}}] = \left\langle a_k^{\mathbf{S}}, \ldots, a_l^{\mathbf{S}} \right\rangle$, gdzie $a_k^{\mathbf{S}} \in A_k^{\mathbf{S}}$, …, $a_l^{\mathbf{S}} \in A_l^{\mathbf{S}}$
w	warunek elementarny postaci $A_i \circ a_i$ lub $A_i \circ A_j$, $\circ$ – symbol relacji w dziedzinie A_i (np. dla dziedzin liczbowych $\circ \in \{=, \neq, <, \leq, >, \geq\}$, warunek elementarny ma wartość logiczną: prawda lub fałsz *Przykłady:* a) Cena > 100,00 zł b) Nazwa = „Książka”
$\tilde{w}$	warunek postaci: $\tilde{w} = w_1 \bullet w_2 \bullet \ldots \bullet w_k$, $w_1, w_2, \ldots w_k$ – warunki elementarne, $\bullet$ – operator logiczny: alternatywa $(\vee)$, koniunkcja $(\wedge)$ lub negacja $(\neg)$ *Przykład:* (Cena > 100,00 zł) $\wedge$ (Cena < 200,00 zł)

2.1.1 OPERATOR RZUTU (PROJEKCJI)

Operacją rzutu (projekcji) relacji $\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$ na zbiorze atrybutów $\tilde{A}^{\mathbf{R}} = A_i^{\mathbf{R}}, \ldots, A_j^{\mathbf{R}}$ nazywamy przekształcenie relacji $\mathbf{R}$ w relację wynikową postaci:

$$\Pi_{\tilde{A}^{\mathbf{R}}} \mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) = \left\{k^{\mathbf{R}}[\tilde{A}^{\mathbf{R}}] : k^{\mathbf{R}} \in \mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)\right\}$$

gdzie: Π – operator rzutowania.

Operacja rzutowania umożliwia utworzenie „pionowego podzbioru" relacji poprzez wybór określonych kolumn.

Przykład:

Dana jest relacja **Towar** o schemacie: **Towar**(NrTowaru, Nazwa, Cena, VAT). Należy wykonać operację rzutowania $\Pi_{\text{Nazwa, Cena}}$ **Towar**(NrTowaru, Nazwa, Cena, VAT).

Towar

NrTowaru	Nazwa	Cena	VAT
1	Książka	75,00 zł	7%
2	Zeszyt	2,00 zł	22%
3	Ołówek	1,50 zł	22%

$\Pi_{\text{Nazwa, Cena}}$ **Towar**

Nazwa	Cena
Książka	7%
Zeszyt	22%
Ołówek	22%

2.1.2 OPERATOR SELEKCJI

Operacją selekcji relacji $\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$ względem warunku $\tilde{w}$ nazywamy przekształcenie relacji $\mathbf{R}$ w relację wynikową postaci:

$$\Sigma_{\tilde{w}}\,\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) = \left\{k^{\mathbf{R}} : k^{\mathbf{R}} \in \mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) \wedge \tilde{w} = prawda\right\}$$

gdzie: Σ – operator selekcji.

Operacja selekcji umożliwia utworzenie „poziomego podzbioru" relacji poprzez wybór krotek spełniających określony warunek.

Przykład:

Dana jest relacja **Towar** o schemacie: **Towar**(NrTowaru, Nazwa, Cena, VAT). Należy wykonać operację selekcji $\Sigma_{\text{VAT=22\%}}$ **Towar**(NrTowaru, Nazwa, Cena, VAT).

Towar

NrTowaru	Nazwa	Cena	VAT
1	Książka	75,00 zł	7%
2	Zeszyt	2,00 zł	22%
3	Ołówek	1,50 zł	22%

$\Sigma_{\text{VAT=22\%}}$ **Towar**

NrTowaru	Nazwa	Cena	VAT
2	Zeszyt	2,00 zł	22%
3	Ołówek	1,50 zł	22%

2.1.3 OPERATOR ZŁĄCZENIA

Operacją złączenia relacji $\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$ i $\mathbf{S}\left(A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}\right)$ względem atrybutów połączeniowych $\tilde{A}^{\mathbf{R}}$ i $\tilde{A}^{\mathbf{S}}$ oraz warunku $\tilde{w}$ nazywamy przekształcenie w relację wynikową postaci:

$$\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) \underset{\tilde{w}}{\otimes} \mathbf{S}\left(A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}\right) =$$

$$\left\{k^{\mathbf{RS}} = \left\langle a_1^{\mathbf{R}}, \ldots, a_n^{\mathbf{R}}, a_1^{\mathbf{S}}, \ldots, a_m^{\mathbf{S}} \right\rangle : \left(k^{\mathbf{R}} \in \mathbf{R}\right) \wedge \left(k^{\mathbf{S}} \in \mathbf{S}\right) \wedge \left(\tilde{w} = prawda\right)\right\}$$

gdzie: $\otimes$ – operator złączenia.

Operacja złączenia polega na scaleniu odpowiednich krotek dwóch różnych relacji pod warunkiem spełnienia warunku logicznego nałożonego na atrybuty połączeniowe.

Przykład:

Dana jest relacja **Faktura** o schemacie **Faktura**(NrFaktury, NrTowaru, IlośćSztuk) oraz relacja **Towar** o schemacie: **Towar**(NrTtowaru, Nazwa, Cena, VAT) . Należy wykonać operację złączenia

Faktura $\underset{\text{NrTowaru} = \text{NrTowaru}}{\otimes}$ **Towar**.

Faktura

NrFaktury	NrTowaru	IlośćSztuk
1	1	10
1	2	5
2	3	3
2	1	5

Towar

NrTowaru	Nazwa	Cena	VAT
1	Książka	75,00 zł	7%
2	Zeszyt	2,00 zł	22%
3	Ołówek	1,50 zł	22%

Faktura $\underset{\text{NrTowaru} = \text{NrTowaru}}{\otimes}$ **Towar**

NrFaktury	NrTowaru	IlośćSztuk	NrTowaru	Nazwa	Cena	VAT
1	1	10	1	Książka	75,00 zł	7%
1	2	5	2	Zeszyt	2,00 zł	22%
2	3	3	3	Ołówek	1,50 zł	22%
2	1	5	1	Książka	75,00 zł	7%

Uwaga: Jeśli operacja złączenia jest wykonywana względem wszystkich wspólnych atrybutów złączanych relacji **R** i **S** oraz warunek połączeniowy jest koniunkcją warunków elementarnych postaci $A_i^{\mathbf{R}} = A_j^{\mathbf{S}}$ to mówimy o złączeniu naturalnym i zapisujemy ten rodzaj złączenia w postaci: $\mathbf{R} \otimes \mathbf{S}$.

2.1.4 OPERATORY TEORIOMNOGOŚCIOWE (ILOCZYN KARTEZJAŃSKI, SUMA, ILOCZYN, RÓŻNICA)

Operator iloczynu kartezjańskiego

Operacją iloczynu kartezjańskiego relacji $\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$ i $\mathbf{S}\left(A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}\right)$ nazywamy przekształcenie w relację wynikową postaci:

$$\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) \times \mathbf{S}\left(A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}\right) =$$
$$\left\{k^{\mathbf{RS}} = \left\langle a_1^{\mathbf{R}}, \ldots, a_n^{\mathbf{R}}, a_1^{\mathbf{S}}, \ldots, a_m^{\mathbf{S}} \right\rangle : \left(k^{\mathbf{R}} \in \mathbf{R}\right) \wedge \left(k^{\mathbf{S}} \in \mathbf{S}\right)\right\}$$

gdzie: $\times$ – operator iloczynu kartezjańskiego.

Operacje sumowania, wyznaczania części wspólnej i różnicy dwóch relacji mogą być wykonywane tylko na relacjach rozpiętych na takich samych schematach. Zasada działania wymienionych operatorów jest analogiczna do znanej z teorii zbiorów np.:: sumowanie polega na utworzeniu relacji złożonej ze wszystkich krotek należących do relacji biorących udział w sumowaniu.

Zapytanie zbudowane w języku algebraicznym jest wyrażeniem w którego skład wchodzą relacje oraz omówione operatory relacyjne. Języki algebraiczne zalicza się do języków proceduralnych gdyż zapytanie jest procedurą podającą w kolejnych krokach sposób generowania odpowiedzi (weź relacje **R** i **S** wykonaj złączenie, następnie wykonaj selekcję względem warunku ..., itd.).

2.2 JĘZYKI OPARTE NA RELACYJNYM RACHUNKU KROTEK I DZIEDZIN

Zapytania zbudowane w językach opartych na relacyjnym rachunku krotek lub dziedzin zbudowane są z dwóch części [2], [4]:

– pierwsza opisuje schemat relacji wynikowej,

– druga podaje warunki, które muszą spełniać krotki tej relacji.

Użytkownik nie podaje więc w tym przypadku procedury postępowania, która będzie prowadziła do uzyskania odpowiedzi. Języki oparte na rachunku relacyjnym są językami nieproceduralnymi. W przypadku zapytań w tego rodzaju językach, system zarządzania bazą danych określa (na podstawie wbudowanych algorytmów) metodę postępowania, która ostatecznie doprowadzi do uzyskania odpowiedzi na określone zapytanie.

Budowa języków opartych na rachunku relacyjnym zostanie omówiona na przykładzie języka opartego na relacyjnym rachunku krotek. W językach tego typu zapytania przedstawiane są w postaci wyrażeń:

$$\{t : f(t)\}$$

gdzie: t – zmienna krotkowa; $f(t)$ – formuła zbudowana z atomów i zestawu operatorów.

2.2.1 ATOM

W formułach występują atomy trzech rodzajów:

- krotka relacji np.: $k^{\mathbf{R}}$ (krotka relacji **R**),
- wyrażenia postaci: $a_i^{\mathbf{R}} \circ a_j^{\mathbf{S}}$ lub $a_i^{\mathbf{R}} \circ a_j^{\mathbf{R}}$ ($a_i^{\mathbf{R}}$ – i-ta składowa krotki relacji **R** (**$a_i^{\mathbf{R}} \in A_i^{\mathbf{R}}$**), $a_j^{\mathbf{S}}$ – j-ta składowa krotki relacji **S** (**$a_j^{\mathbf{S}} \in A_j^{\mathbf{S}}$**), $\circ$ – symbol relacji w dziedzinie $A_i^{\mathbf{R}}$ i $A_j^{\mathbf{S}}$ (dziedziny muszą być tych samych typów; liczbowe, tekstowe itp.)),
- wyrażenia $a_i^{\mathbf{R}} \circ c$ lub $c \circ a_i^{\mathbf{R}}$ (c – stała należąca do dziedziny $A_i^{\mathbf{R}}$).

2.2.2 OPERATORY

- operatory porównania arytmetycznego: =, ≠, <, ≤, >, ≥
- kwantyfikatory: ∃ (istnieje) oraz ∀ (dla każdego)
- operatory logiczne: negacja $(\neg)$, koniunkcja $(\wedge)$ oraz alternatywa $(\vee)$

Priorytety operatorów są zgodne z podaną kolejnością, najwyższy priorytet mają operatory arytmetyczne, najniższy alternatywa. Jeżeli jest to konieczne elementy formuły trzeba ujmować w nawiasy.

Język oparty na relacyjnym rachunku krotek jest równoważny językowi algebraicznemu. Równoważność ta oznacza, że wszystko co może być wyrażone w jednym języku można również wyrazić w drugim.

2.2.3 PRZYKŁAD

Należy przedstawić operacje języka algebraicznego za pomocą wyrażeń rachunku relacyjnego krotek.

$\Pi_{A_{i_1}^{\mathbf{R}},\ldots,A_{i_k}^{\mathbf{R}}} \mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$ $i_1, \ldots, i_k \in [1, n]$	$\left\{t : \left(\exists k^{\mathbf{R}}\right)\left(\mathbf{R}\left(k^{\mathbf{R}}\right) \wedge \left(t_1 = a_{i_1}^{\mathbf{R}}\right) \wedge \ldots \wedge \left(t_k = a_{i_k}^{\mathbf{R}}\right)\right)\right\}$ t_j – j-ty element krotki t $j = 1, \ldots, k$
$\Sigma_{\tilde{w}} \mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$	$\left\{t : \mathbf{R}(t) \wedge \tilde{w}'\right\}$, gdzie formuła $\tilde{w}'$ odpowiada warunkowi $\tilde{w}$ po zastąpieniu składowych krotki $k^{\mathbf{R}}$ elementami zmiennej krotkowej t
$\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) \underset{\tilde{w}}{\otimes}$ $\mathbf{S}\left(A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}\right)$	$\left\{t : \left(\exists k^{\mathbf{R}}\right)\left(\exists k^{\mathbf{S}}\right)\left(\mathbf{R}\left(k^{\mathbf{R}}\right) \wedge \mathbf{S}\left(k^{\mathbf{S}}\right) \wedge \tilde{w}'\right)\right\}$, gdzie $\tilde{w}'$ jak powyżej
$\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) \times$ $\mathbf{S}\left(A_1^{\mathbf{S}}, A_2^{\mathbf{S}}, \ldots, A_m^{\mathbf{S}}\right)$	$\left\{t : \left(\exists k^{\mathbf{R}}\right)\left(\exists k^{\mathbf{S}}\right)\left(\mathbf{R}\left(k^{\mathbf{R}}\right) \wedge \mathbf{S}\left(k^{\mathbf{S}}\right) \wedge \left(t_1 = a_1^{\mathbf{R}}\right) \wedge \ldots \wedge \left(t_n = a_n^{\mathbf{R}}\right) \wedge \left(t_{n+1} = a_1^{\mathbf{S}}\right) \wedge \ldots \wedge \left(t_{n+m} = a_m^{\mathbf{S}}\right)\right)\right\}$
$\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) \cup$ $\mathbf{S}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$	$\left\{t : \mathbf{R}(t) \vee \mathbf{S}(t)\right\}$
$\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) \cap$ $\mathbf{S}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$	$\left\{t : \mathbf{R}(t) \wedge \mathbf{S}(t)\right\}$
$\mathbf{R}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right) -$ $\mathbf{S}\left(A_1^{\mathbf{R}}, A_2^{\mathbf{R}}, \ldots, A_n^{\mathbf{R}}\right)$	$\left\{t : \mathbf{R}(t) \wedge \neg \mathbf{S}(t)\right\}$

Języki oparte na relacyjnym rachunku dziedzin różnią się od języków rachunku relacyjnego krotek definicją atomu. W tym przypadku atom jest konstruowany nie ze zmiennych krotkowych ale zmiennych dziedzinowych. Do tworzenia formuł wykorzystywany jest ten sam zestaw operatorów.

Dowody równoważności omówionych trzech języków abstrakcyjnych można znaleźć w pozycji [4].

3. PRZYKŁADY

W bieżącym punkcie zostaną przedstawione przykładowe kwerendy projektu „Biblioteka" (omawianego w ćwiczeniu poprzednim, którego tematem było projektowanie struktury baz danych). Zapytania te zostaną przedstawione w trzech językach:

- abstrakcyjnym języku algebraicznym,

oraz popularnych językach wykorzystywanych w rzeczywistych systemach zarządzania bazami danych:

- QBE,
- SQL.

Baza „Biblioteka" po wykonaniu normalizacji zawierała następujące relacje w III FN:

Książki(<u>IDKS</u>, Tytuł, Autor),
Wypożyczający(<u>IDW</u>, Nazwisko, Adres, IDKT),
Kategoria(<u>IDKT</u>, NazwaKT, Termin, Ilość),
Wypożyczenia(<u>IDKS</u>, <u>DataW</u>, IDW, DataZ).

Należy zaprojektować trzy kwerendy:

1) Zadaniem pierwszej kwerendy jest wylistowanie wszystkich wypożyczających mieszkających w Zielonej Górze. Lista ta powinna zawierać nazwiska znalezionych osób.

2) Druga kwerenda powinna wyświetlać wszystkie książki wypożyczone określonego dnia np.: 2001-09-01. Należy wyświetlać tytuł i autora każdej wypożyczonej książki.
3) Zadaniem trzeciej kwerendy jest wyświetlenie książek, które aktualnie są wypożyczone. Lista powinna zawierać nazwisko i adres wypożyczającego wraz z tytułem, autorem i datą wypożyczenia książki.

3.1 JĘZYK ALGEBRAICZNY

Zapytanie pierwsze jest najprostszym z rozważanych zapytań wymaga jedynie wykonania selekcji i rzutu działających tylko na pojedynczej relacji: Wypożyczający:

$$\Pi_{\text{Nazwisko}} (\Sigma_{\text{Adres=”Zielona Góra”}} \text{Wypożyczający}).$$

Bardziej skomplikowane rodzaje zapytań wymagają użycia złączenia (najczęściej naturalnego), a następnie ewentualnie selekcji i/lub rzutu. W przypadku kwerendy drugiej należy złączyć relacje Wypożyczenia i Książki (atrybut IDKS w relacji Wypożyczenia odpowiada atrybutowi IDKS w relacji Książki – złączenie jest więc złączeniem naturalnym), a następnie wykonać selekcję (wybrać tylko te książki, które były wypożyczone 2001-09-01) i rzutowanie (wybrać kolumny: Tytuł i Autor z powstałej relacji):

$$\Pi_{\text{Tytuł, Autor}} (\Sigma_{\text{DataW=2001-09-01}}(\text{Wypożyczenia} \otimes \text{Książki})).$$

Kwerenda ta może być również zapisana w sposób bardziej optymalny (wymagający mniej obliczeń) następująco:

$$\Pi_{\text{Tytuł, Autor}} (\Pi_{\text{IDKS}} (\Sigma_{\text{DataW=2001-09-01}}(\text{Wypożyczenia})) \otimes \text{Książki})).$$

W tym przypadku złączenia naturalne wykonywane jest na zbiorze tylko tych książek, które będą wyświetlane a nie tak jak w przypadku poprzednim na zbiorze wszystkich książek.

Najbardziej złożonym zapytaniem jest zapytanie trzecie. Wymaga ono złączenia aż trzech relacji: Wypożyczający, Książki oraz Wypożyczenia, selekcji – w celu wyboru książek aktualnie wypożyczonych (DataZ jest pusta) oraz projekcji (wybór kolumn Nazwisko, Adres, Tytuł, Autor,

DataW). Kwerenda ta może być, podobnie jak poprzednia, przedstawiona na kilka sposobów, np.:

$\Pi_{\text{Nazwisko, Adres, Tytuł, Autor, DataW}}$

$(((\Sigma_{\text{DataZ jest pusta}}(\text{Wypożyczenia})) \otimes \text{Książki}) \otimes \text{Wypożyczający})$.

3.2 JĘZYK SQL

Język SQL (Structured Query Language – strukturalny język zapytań) powstał w firmie IBM pod koniec lat siedemdziesiątych. Ma on cechy zarówno języka algebraicznego jaki i języka opartego na rachunku krotek [1], [2], [4]. Podstawową formą zapytania jest wyrażenie:

SELECT $\mathbf{R}_{i_1}.A_1,\ldots,\mathbf{R}_{i_k}.A_k$ lub *
FROM $\mathbf{R}_1,\ldots,\mathbf{R}_j$
WHERE $\tilde{w}$

gdzie: $\mathbf{R}_{i_1}.A_1,\ldots,\mathbf{R}_{i_k}.A_k$ – atrybuty relacji wynikowej (jeżeli atrybut A_l występuje tylko w jednej z relacji $\mathbf{R}_1,\ldots,\mathbf{R}_j$ to część opisującą relację z której pochodzi atrybut można pominąć); symbol * może wystąpić zamiast wyrażenia $\mathbf{R}_{i_1}.A_1,\ldots,\mathbf{R}_{i_k}.A_k$ – wskazuje, że należy pobrać wszystkie atrybuty ze wszystkich relacji składowych $\mathbf{R}_1,\ldots,\mathbf{R}_j$; $i_1,\ldots,i_k \in [1,\ j]$; $\mathbf{R}_1,\ldots,\mathbf{R}_j$ – relacje których atrybuty są wykorzystywane w kwerendzie; $\tilde{w}$ – warunek logiczny.

Pierwsze z rozważanych zapytań, zgodnie z powyżej przedstawioną składnią, można zapisać:

```
SELECT Nazwisko
FROM Wypożyczający
WHERE Adres = „Zielona Góra”
```

Druga kwerenda wymaga użycia złączenia naturalnego relacji Wypożyczenia i Książki. Złączenie to może być zrealizowane poprzez odpowiednią postać warunku $\tilde{w}$. Warunek ten będzie więc opisywał sposób łączenia „wiązanych” relacji oraz będzie zawierał dodatkowe

ograniczenie na datę wypożyczenia (2001-09-01). Obydwa warunki elementarne będą połączone operatorem koniunkcji. W języku SQL zostały zdefiniowane następujące operatory logiczne: koniunkcji AND, alternatywy OR oraz negacji NOT. Ostatecznie przykładowe zapytanie przedstawia wyrażenie:

```
SELECT Tytuł, Autor
FROM Wypożyczenia, Książki
WHERE  (Wypożyczenia.IDKS=Książki.IDKS) AND
       (DataW=2001-09-01)
```

Trzecie z rozważanych zapytań powinno generować listę wszystkich aktualnie wypożyczonych książek. Wymaga to sprawdzenia czy data zwrotu książki ma wartość pustą. Warunek sprawdzający pustą wartość przykładowego atrybutu A przedstawia wyrażenie: A IS NULL (jeżeli A nie przypisano wartości, wyrażenie ma wartość logiczną prawda). Stąd zapytanie trzecie można zapisać jako:

```
SELECT Nazwisko, Adres, Tytuł, Autor, DataW
FROM Wypożyczenia, Książki, Wypożyczający
WHERE  (Wypożyczenia.IDKS=Książki.IDKS) AND
       (Wypożyczenia.IDW=Wypożyczający.IDW) AND
       (DataZ IS NULL)
```

W poleceniu SELECT można również wymuszać określony sposób porządkowania danych relacji wynikowej. Służy do tego klauzula ORDER BY. Na przykład, w celu wyświetlenia alfabetycznej listy książek można sformułować zapytanie:

```
SELECT Tytuł, Autor
FROM Książki
ORDER BY Tytuł
```

Język SQL pozwala również na wykonywanie obliczeń udostępniając operatory agregujące: SUM (suma), AVG (średnia), MAX (maksimum), MIN (minimum) oraz COUNT (liczność). Dzięki temu możliwe jest na przykład na obliczenie ilości wypożyczonych książek przez wypożyczającego o identyfikatorze 123:

```
SELECT COUNT(IDKS)
FROM Wypożyczenia
WHERE  IDW=123
```

Dodatkowo obliczenia te mogą być również przeprowadzane na grupach krotek, a nie na wszystkich krotkach danej relacji (dany zbiór krotek stanowi grupę, jeżeli wartości we wskazanym polu tych krotek są takie same). Można na przykład wyliczyć ilość wypożyczanych książek dla każdego z wypożyczających. Należy w tym przypadku podzielić dane z relacji Wypożyczenia na grupy na podstawie identyfikatorów wypożyczających, a następnie w każdej z grup należy zliczyć ilość krotek. W tym przypadku należy skorzystać z nieco bardziej rozbudowanej wersji polecenia SELECT z dodatkową klauzulą GROUP BY (po poleceniu GROUP BY należy wskazać atrybuty relacji według których przeprowadzane będzie grupowanie krotek):

```
SELECT IDW, COUNT(IDKS)
FROM Wypożyczenia
GROUP BY IDW
```

Język SQL oprócz możliwości wyszukiwania informacji (przy pomocy polecenia SELECT) posiada również typowe dla języków DML mechanizmy umożliwiające modyfikację zawartości bazy danych:

- dołączanie nowych krotek do określonej relacji jest możliwe dzięki poleceniu INSERT:

  ```
  INSERT INTO Wypożyczenia (IDW, IDKS, DataW)
  VALUES (123, 2222, 2001-09-01),
  ```

- aktualizację danych umożliwia polecenie UPDATE:

  ```
  UPDATE Wypożyczający
  SET Adres = „Nowa Sól”
  WHERE  IDW=123
  ```

- usuwanie danych polecenie DELETE:

  ```
  DELETE FROM Wypożyczający
  WHERE  IDW=123
  ```

Przykładowe polecenia powodują odpowiednio:

- dodanie do relacji Wypożyczenia nowej krotki opisującej fakt wypożyczenia 01 września 2001 roku książki o sygnaturze 2222 przez wypożyczającego o identyfikatorze 123
- aktualizację adresu wypożyczającego o identyfikatorze 123 („Nowa Sól")
- usunięcie z relacji Wypożyczający wypożyczającego o identyfikatorze 123.

Wymienione polecenia nie wyczerpują wszystkich możliwości języka SQL, który oprócz części DML posiada również zestaw instrukcji charakterystycznych dla języków opisu schematów danych (DDL – Data Description Languages).

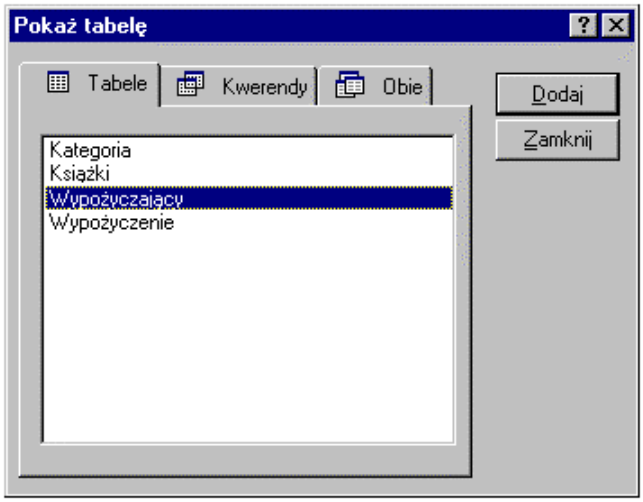
3.3 JĘZYK QBE

Język QBE (Query By Example – zapytanie przez przykład) posiada cechy języka opartego na rachunku dziedzin [2], [4]. Ze względu na prostotę tworzenia kwerend jest on bardzo popularnym językiem relacyjnych baz danych. Sposób projektowania zapytań w języku QBE zostanie omówiony na przykładzie implementacji tego języka w środowisku Microsoft Access [3].

Projektowanie kwerend, podobnie jak projektowanie tabel, odbywa się w specjalnym oknie - oknie projektu kwerendy (*Rys. 1*). Aby je otworzyć należy w oknie bazy danych wybrać zakładkę **Kwerendy**, a następnie przycisk **Nowy** (uruchamianie programu MS Access oraz elementy projektu bazy danych w tym systemie zostały omówione w ćwiczeniu poprzednim). Kwerendę można zaprojektować korzystając z kilku proponowanych przez środowisko możliwości. W ćwiczeniu omówiona została omówiona tylko jedna z nich – przy użyciu tzw. **Widoku projekt**. Ten sposób wymaga formułowania kwerendy od podstaw i nie udostępnia narzędzi automatyzujących ten proces.

Po zaakceptowaniu wyboru opcji **Widok projekt** na ekranie pojawia się okno projektu kwerendy (*Rys. 2*) oraz okno dialogowe **Pokaż tabelę** (*Rys. 1*). Okno **Pokaż tabelę** umożliwia wybór relacji lub kwerend źródłowych zawierających dane potrzebne do konstrukcji nowej

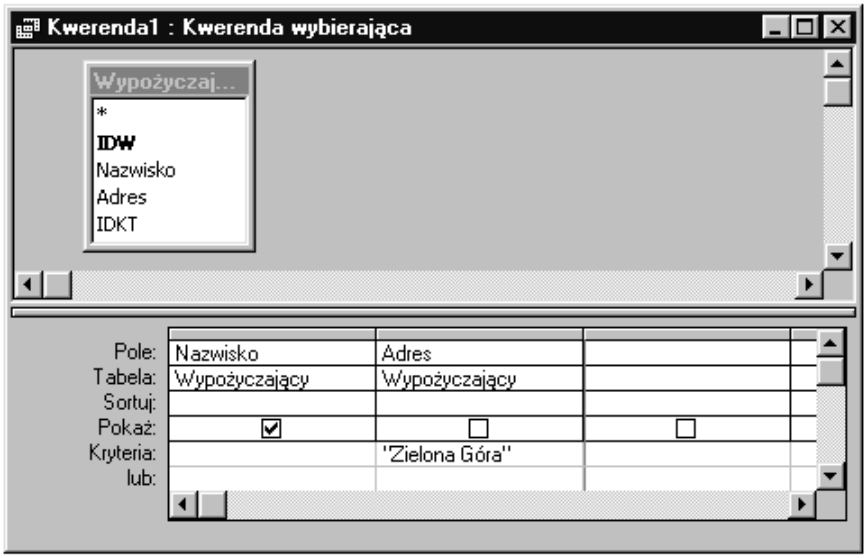
kwerendy. Okno **Pokaż tabelę** zawiera trzy zakładki umożliwiające wybór typu widocznych obiektów. Domyślnie wybrana jest zakładka **Tabele** i tylko te obiekty są widoczne po otwarciu okna (*Rys. 1*). Przełączenie na inną zakładkę pozwala na przeglądanie stworzonych już kwerend (zakładka **Kwerendy**) lub obu rodzajów obiektów jednocześnie (zakładka **Obie**). Wyboru obiektów źródłowych dokonuje się przez wskazanie ich na liście i kliknięcie na przycisku **Dodaj**. Wybrany obiekt pojawia się w oknie projektu kwerendy (*Rys. 2*). Po dodaniu wszystkich obiektów źródłowych okno dialogowe należy zamknąć klikając na przycisku **Zamknij**.



Rys. 1 Okno dialogowe Pokaż tabelę

Po zamknięciu okno to, w razie potrzeby, może być przywołane na dowolnym etapie projektowania kwerendy przez wybór opcji **Pokaż tabelę** z menu **Kwerenda**.

Okno projektu kwerendy składa się z dwóch części rozdzielonych poziomą linią (*Rys. 2*). Część górna to obszar przeznaczony na obiekty źródłowe. Znajdują się w niej wszystkie tabele i kwerendy, które zostały dodane do projektu za pomocą okna **Pokaż tabelę**. Część dolna to tzw. szablon QBE, czyli arkusz, w którym należy zbudować wzorzec odpowiedzi. Pojedyncza kolumna tego arkusza odpowiada jednej kolumnie relacji wynikowej, będącej rezultatem działania kwerendy.



Rys. 2 Okno projektu kwerendy

Pierwszy górny wiersz zatytułowany **Pole** przeznaczony jest na nazwę wybranego pola obiektu źródłowego. Rubryka wyposażona jest w rozwijaną listę nazw, z której należy dokonać wyboru pola mającego pojawić się w odpowiedzi. Wiersz **Tabela** informuje z którego obiektu źródłowego pochodzi dane pole. Wyświetlane w wyniku działania kwerendy informacje mogą być sortowane w wybranym porządku, który powinien zostać ustalony w wierszu **Sortuj**. Możliwe ustawienia to rosnąco, malejąco lub bez sortowania. Wyboru dokonuje się korzystając z rozwijalnej listy umieszczonej w rubryce **sortuj**. Jako kryterium sortowania program wybiera tylko te pola, które mają wybraną opcję rosnąco lub malejąco. Przy ustalaniu porządku sortowanych danych istotna jest kolejność pól umieszczonych w szablonie QBE. Wyświetlane rekordy są sortowane przede wszystkim po danych pochodzących z pola umieszczonego jako pierwsze z lewej strony. Dopiero kiedy wartości tego pola dla dwóch lub więcej rekordów są identyczne, brane jest pod uwagę pole kolejne. Następne pola będą uwzględnione jedynie kiedy dwa pierwsze będą miały te same wartości w dwóch lub więcej rekordach, itd. Kolejnym wierszem szablonu QBE jest wiersz **Pokaż.** Zawiera on jedynie pole wyboru, które umożliwia na ustalenie, czy dana kolumna będzie widoczne w arkuszu wynikowym. Ostatnie wiersze

szablonu, z których dwa pierwsze nazwano **Kryteria** oraz **lub**, przeznaczone są na ustalanie dodatkowych kryteriów wyboru danych wyświetlanych jako odpowiedź.

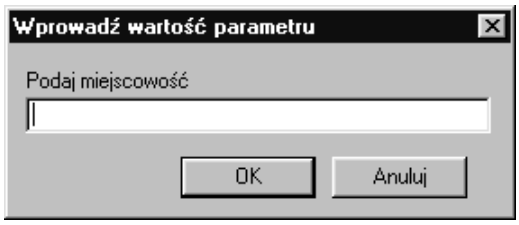
Utworzoną kwerendę można uruchomić i sprawdzić jej działanie po zamknięciu okna projektu, z poziomu okna bazy danych, wybierając przycisk **Otwórz**. Ten sam efekt można osiągnąć niezamykając okna projektu kwerendy, wybierając opcję **Uruchom** z menu **Kwerenda**. W obydwu przypadkach efektem będzie odpowiedź w formie standardowego arkusza zawierającego dane zgodne ze strukturą ustaloną w arkuszu QBE.

Poniżej przedstawione zostaną projekty trzech rozważanych zapytań, tym razem wyrażonych w języku QBE.

Pierwsza kwerenda jest najprostszą spośród omawianych kwerend. Jej zadaniem jest wylistowanie wszystkich wypożyczających mieszkających w Zielonej Górze. Formułowanie nowej kwerendy należy rozpocząć od dodania do nowootwartego projektu obiektów zawierających potrzebne dane. W przypadku omawianej kwerendy całość interesujących informacji zawiera tabela „Wypożyczający". Po dodaniu tabeli należy określić, które pola będą widoczne w arkuszu odpowiedzi. W tym przypadku jest to tylko pole *Nazwisko*. Ostatnią czynnością jest ustalenie warunku pozwalającego na wyświetlenie wypożyczających mieszkających w Zielonej Górze. Aby ograniczyć wyświetlane informacje do osób spełniających pewien warunek należy wprowadzić odpowiednie wyrażenie w wierszu **Kryteria**. Wiersz ten służy do określenia warunków, które ma spełniać rekord, aby został wyświetlony w wyniku działania kwerendy. Kryteria mogą być bardzo złożone i dotyczyć jednocześnie kilku kolumn arkusza QBE. Można łączyć je używając koniunkcji, lub alternatywy i wpisując w odpowiednim wierszu arkusza (wiersz **Kryteria** oraz wiersz **lub**). Kryterium należy wprowadzić w kolumnie zawierającej pole, którego ono dotyczy. Tworząc je można stosować standardowe operatory relacyjne (<, >, <=, >=. =) i logiczne (And, Or, Not). W przypadku omawianej kwerendy listę osób mieszkających w Zielonej Górze można uzyskać wprowadzając w polu **Kryteria** kolumny *Adres* wyrażenie: *="Zielona Góra"*. Wymaga to oczywiście dodania do arkusza QBE kolumny *Adres*, która ze względu na to, że nie powinna być widoczna po uruchomieniu

kwerendy, ma odznaczony element w wierszu **Pokaż** (kwerenda ta została przedstawiona na *Rys. 2*).

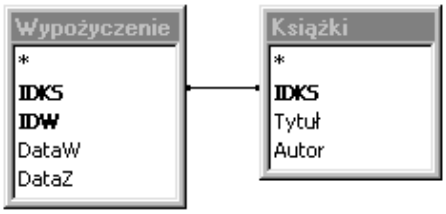
Uwaga! MS Access umożliwia tworzenie sparametryzowanych kwerend. Oznacza to, że kwerenda przed wyświetleniem relacji wynikowej prosi użytkownika o podanie wartości wszystkich parametrów. Parametry takie można definiować wprowadzając w wierszu kryterium tekst umieszczony w nawiasach prostokątnych. W omawianej kwerendzie kryterium w kolumnie Adres może wyglądać następująco: *=[Podaj miejscowość]*. Wprowadzenie takiego wyrażenia spowoduje wyświetlenie okna dialogowego (*Rys. 3*) przy każdym uruchomieniu kwerendy. Wartość wpisana w polu edycji okna zostanie podstawiona w miejsce napisu [Podaj miejscowość] i to dopiero ostatecznie ustali postać warunku w kryterium. Tak skonstruowana kwerenda będzie więc wyświetlać osoby, które mieszkają w podanej przez użytkownika miejscowości.



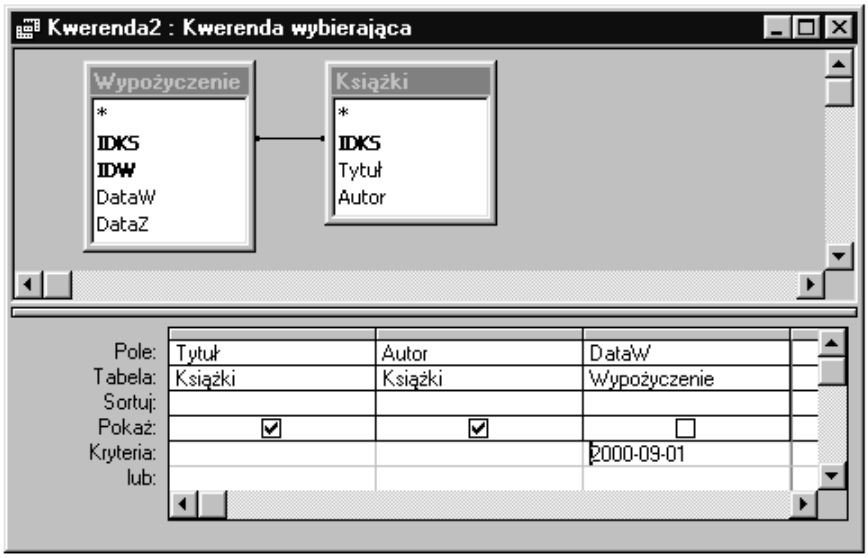
Rys.3 Okno dialogowe z pytaniem o wartość parametru kwerendy

Druga kwerenda powinna wyświetlać wszystkie książki wypożyczone określonego dnia np.: 2001-09-01. Tak jak poprzednio należy w pierwszej kolejności ustalić, które z obiektów bazy danych zawierają interesujące informacje. W tym przypadku są to dwie tabele: Książki oraz Wypożyczenie. W kwerendach zawierających kilka obiektów źródłowych bardzo istotną rolę odgrywają sprzężenia. Sprzężenie odpowiada operatorowi relacyjnemu operatorowi złączenia. W przypadku omawianej kwerendy należy złączyć relacje Wypożyczenia i Książki (atrybut IDKS w tabel Wypożyczenia odpowiada atrybutowi IDKS w tabeli Książki). W przypadku opisywanych kwerend, po dodaniu tabel źródłowych, Access automatycznie wprowadza sprzężenie pomiędzy polem IDKS z tabeli Książki i polem o takiej samej nazwie z tabeli Wypożyczenie (Access automatycznie wiąże pola o tych samych nazwach). Jeżeli nazwy pól nie byłyby identyczne sprzężenie nie

zostanie automatycznie dodane. W takim przypadku użytkownik musi sam określić sposób łączenia obiektów źródłowych. Dodanie nowego sprzężenia odbywa się przez przeciągnięcie kursora myszy, przy wciśniętym lewym przycisku, od wybranego pola z obiektu pierwszego do łączonego pola obiektu drugiego. Przykładowo, aby ustalić sprzężenie pomiędzy polami IDKS w obu tabelach źródłowych, należy ustawić kursor myszy na polu IDKS w tabeli Książki, wcisnąć lewy przycisk myszy i trzymając go przeciągnąć kursor na pole IDKS w tabeli Wypożyczenie. Po wykonaniu tej operacji pojawi się sprzężenie pokazane na *Rys. 4*.



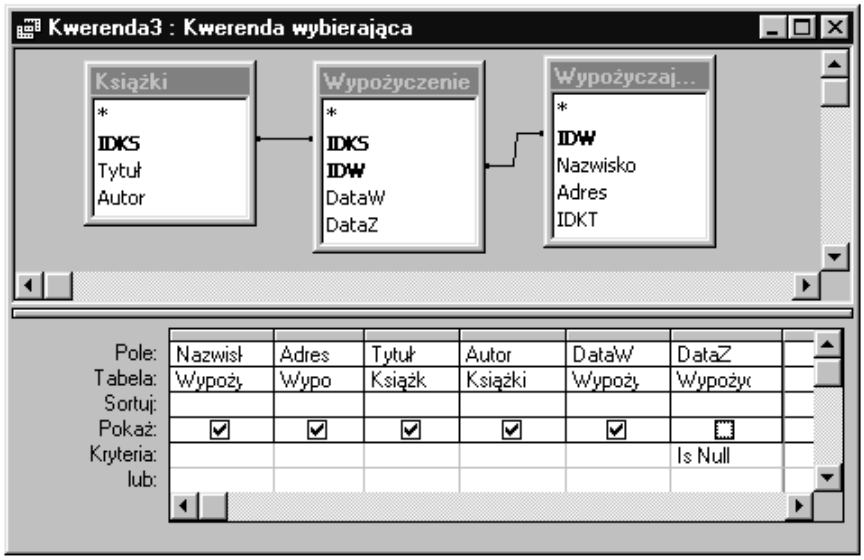
Rys. 4 Sprzężenie obiektów źródłowych w kwerendzie



Rys. 5 Projekt kwerendy nr 2

Kolejnym krokiem jest określenie pól, które będą widoczne w wyniku wykonania kwerendy (Tytuł i Autor z tabeli Książki) oraz określenie warunków określających, które rekordy powinny być pokazane w wyniku uruchomienia kwerendy (DataW = 2001-09-01). Ostatecznie projekt kwerendy drugiej przedstawia *Rys. 5*.

W celu zaprojektowania kwerendy trzeciej wymagana jest dodatkowa umiejętność zapisu warunku, który opisywałby sytuację braku wartości w określonym polu bazy danych (w tym przypadku pole DataZ tabeli Wypożyczenia jest puste). Podobnie jak w przypadku języka SQL wyrażeniem sprawdzającym czy dane pole nie zawiera żadnych wartości jest *Is Null*. Gotowy projekt kwerendy trzeciej przedstawiono na poniższym rysunku.



Rys. 6 Projekt kwerendy nr 3

Język QBE pozwala podobnie jak SQL na wykonywanie obliczeń udostępniając funkcje agregujące: Suma, Średnia, Maksimum, Minimum oraz Zlicz. Funkcje te stają się dostępne po dodaniu dodatkowego wiersza w arkuszu QBE. Można to zrobić wybierając polecenie **Podsumowania** z menu **Widok.** Po wykonaniu powyższej czynności w arkuszu pojawi się wiersz **Podsumowania**. Operacje agregujące mogą być, podobnie jak w SQL, wykonywane są na określonych grupach

rekordów. Przykłady kwerend wykorzystujących funkcje agregujące przedstawiają dwa kolejne rysunki.

Rys. 7 Projekt kwerendy zliczającej książki wypożyczane przez wypożyczającego o identyfikatorze 123

Rys. 8 Projekt kwerendy zliczającej wypożyczane książki dla każdego wypożyczającego

W języku QBE możliwa jest również modyfikacja zawartości bazy danych (dodawanie rekordów, usuwanie, aktualizacja) przy pomocy kwerend. W takim przypadku należy jednak zmienić typ kwerendy z kwerendy wybierającej (domyślna) na kwerendę dołączającą, usuwającą lub aktualizującą. Więcej informacji o tego typu kwerendach można znaleźć w plikach pomocy programu MS Access i literaturze uzupełniającej.

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

Podstawowe informacje dotyczące uruchomienia programu MS Accesss zostały omówione w ćwiczeniu poprzednim, którego tematem było projektowanie struktury baz danych.

6. ZADANIA DO WYKONANIA

1. Zaprojektuj kwerendy wyszukujące w bazie dane wypożyczających po różnie dobranych kryteriach: nazwisko, adres, data wypożyczenia, itp.
2. Zaprojektuj kwerendę wyświetlającą listę czytelników z podanego przedziału alfabetycznego. Początek i koniec przedziału powinien być podawany z klawiatury w chwili uruchamiania kwerendy.
3. Zaprojektuj kwerendy obliczające średnią i maksymalną liczbę wypożyczonych książek dla wszystkich czytelników.
4. Zaprojektuj kwerendę wyświetlającą nazwiska czytelników, których liczba wypożyczeń jest równa maksymalnej liczbie wypożyczeń. Należy skorzystać z kwerendy zaprojektowanej w zadaniu 3.
5. Kwerendy z zadania 3. zaprojektuj tak, aby wykonywały obliczenia w rozbiciu na kategorie wypożyczających.

6. Zaprojektuj kwerendy usuwające książkę o podanym identyfikatorze z tabeli Książki oraz odwołania do niej w tabeli Wypożyczenie. Uwaga: kwerendy tego typu nie mogą być zrealizowane jako kwerendy wybierające.
7. Zaprojektuj kwerendę wyświetlającą listę aktualnie niewypożyczonych książek. Uwaga: rozwiązanie tego zadania nie jest możliwe przy pomocy jednej kwerendy.

LITERATURA

[1] Gruber Martin, SQL, Wydawnictwo HELION, Gliwice, 1996.

[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.

[5] Codd E.F., Relational completeness of data base sublanguages, Ibidem, 65-98.