ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1993

Seria: INFORMATYKA z. 22 Nr kol. 1195

Małgorzata BACH
Jerzy GRZYWOCZ

PORÓWNANIE MIKROKOMPUTEROWYCH SYSTEMÓW BAZ DANYCH

Streszczenie. Artykuł zawiera porównanie popularnych systemów zarządzania relacyjnymi bazami danych. Jako kryterium przyjęto czas rozwiązywania testowych zadań wyszukiwania. Badano również wpływ poziomu języka zapytań i rozmiarów zadań wyszukiwania na czas ich rozwiązywania.

A COMPARISON OF DATABASE SYSTEMS FOR PERSONAL COMPUTERS

Summary. This paper contains a coparison of popular relational database management systems. A time of answering a query was assumed as a criterion. Also, an influence of a query language level, and of a size of queries on a time of answering a query was tested.

COMPARAISON DE SYSTÈMES DE BASES DE DONNÉES POUR MICRO-ORDINATEUR

Résumé. L'article contient la comparaison de systèmes populaires de gestion de bases de données relationnelles. Le temps de réponse aux requêtes d'essai de recherche est le critère de performance. On a testé aussi l'influence de niveau du language d'interrogation et de taille de la tâche de recherche sur le temps de réalisation de recherche.

1. Wstęp

Jednym z elementów decydujących o nastawieniu użytkownika do systemu zarządzania bazą danych jest czas rozwiązywania zadania wyszukiwania, czyli czas upływający od momentu podania systemowi zadania wyszukiwania do chwili uzyskania wszystkich wyszukiwanych danych we wskazanym miejscu.

Celem pracy jest porównanie wybranych systemów mikrokomputerowych, relacyjnych baz danych. Jako kryterium przyjęto czas rozwiązywania dwóch testowych zadań wyszukiwania w środowisku jednakowym dla wszystkich badanych systemów. Testowano również wpływ poziomu języka zapytań i złożoności obliczeniowej zadań wyszukiwania na czas ich rozwiązywania. Badane systemy były zainstalowane na pojedynczym stanowisku.

Zakres prac obejmował :

- zapisanie każdego z zadań jako programu w j. zapytań, czyli programu wyszukiwania dla wszystkich badanych systemów; dla każdego badanego systemu napisano, używając różnych konstrukcji językowych, kilka wariantów programu odpowiadających semantycznie jednemu zadaniu wyszukiwania;
- zmierzenie czasu wykonania każdego z programów dla dwóch wariantów bazy danych różnej wielkości;
- zbadanie wpływu poziomu języka zapytań, zastosowanych konstrukcji językowych i rozmiaru zadania na efektywość systemu; porównanie badanych systemów pod względem efektywności.

Do badań wybrano systemy relacyjnych baz danych, których języki zapytań odpowiadają językowi systemu dBASE, a dane przechowywane są w plikach typu DBF . Dla porównania zbadano system INFORMIX SQL.

Należy zaznaczyć, że system INFORMIX SQL, w przeciwieństwie do pozostałych badanych systemów przeznaczonych dla systemu operacyjnego DOS, jest przeznaczony dla systemu UNIX (i pochodnych). Wersja INFORMIX'a działająca w systemie DOS musi współpracować z dodatkowym oprogramowaniem, co zmniejsza szybkość działania systemu INFORMIX.

Zbadano następujące systemy :

- dBASE III Plus,
- dBASE IV wersja 1.0,
- Clipper wersja Summer '87,
- Clipper wersja 5.0,
- FoxBASE Plus wersja 2.0,
- FoxPro wersja 1.02,

- FoxPro wersja 2.0,
- Informix wersja 2.0 .

2. Relacyjna baza danych. Pojęcia podstawowe

Relacyjna baza danych może być zdefiniowana jako zbiór relacji, czyli dwuwymiarowych tablic. Kolumna takiej tablicy to **atrybut**, natomiast wiersz nosi nazwę **krotka**. Każda relacja ma swoją nazwę. Przykładowo, do pomiarów użyto następującej bazy danych o nazwie PROJEKTY.

MAGAZYN

NRM	NRC	ILOSC
M1	C2	50
...	...	...

DCP

NRD	NRC	NRP	ILOSC
D1	C5	P8	10
...	...	...	...

BIURO_PR

NRB	NRP
B1	P3
...	...

Rys. 1. Baza danych PROJEKTY
Fig. 1. Database PROJECTS

Każda krotka relacji MAGAZYN informuje, że w magazynie o numerze NRM znajduje się część o numerze NRC w ilości ILOSC. Krotki relacji DCP zawierają informację o tym, że dostawca o numerze NRD dostarcza część o numerze NRC do projektu o numerze NRP w ilości ILOSC. W każdej krotce relacji BIURO_PR jest informacja, że w biurze o numerze NRB realizowany jest projekt o numerze NRP.

Zbiór atrybutów relacji tworzy **schemat relacji**. Zapis r(R) oznacza relację r o schemacie R, np. BIURO_PR(NRB, NRP).
Zapis t[A] oznacza wartość atrybutu A w krotce t.

System zarządzania bazą danych to zestaw środków programowych umożliwiających użytkownikom operowanie na danych i pozwalających administratorowi bazy danych na centralne sterowanie danymi.

Do opisu operacji na relacjach wprowadzono zbiór operatorów tworzących **algebrę relacji**. Najczęściej używa się podzbioru tych operatorów.
W jego skład wchodzą

- selekcja,
- projekcja,
- łączenie.

Operacja selekcji wykonana na relacji r(R) tworzy nową relację o schemacie R zawierającą te krotki z relacji r, w których spełniony jest warunek selekcji. Operację selekcji dla przykładowego warunku selekcji : wartość atrybutu A wynosi a , można zdefiniować następująco :

$$\sigma_{A=a}(r) = r'(R) = \{ t \in r \mid t[A] = a \}$$

W powyższym zapisie

r - relacja o schemacie R,

A - atrybut należący do R,

a - wartość atrybutu A.

Operacja projekcji tworzy relację r' o schemacie X, zawierającą te krotki t[X], które uprzednio należały do relacji r. Operację projekcji można zdefiniować następująco :

$$\pi_X(r) = r'(X) = \{ t[X] \mid t \in r \}$$

W powyższym zapisie

r - rzutowana relacja,

X - podzbiór schematu relacji R,

$\pi_X(r)$ - rzut (czyli projekcja) r na X.

Operator łączenia działa na dwóch relacjach r(R) i s(S). Istnieje łączenie naturalne i θ-łączenie. Autorzy używali operatora θ-łączenia. Powoduje on utworzenie relacji wynikowej q(Q) zawierającej te wszystkie kombinacje połączeń krotek z relacji r i s, dla których spełniony jest warunek nałożony na wartości wybranych atrybutów.
Zapis formalny operacji łączenia jest następujący:

$$q(Q) = r \underset{\langle WŁ \rangle}{\bowtie} s = \sigma_{\langle WŁ \rangle} (r \times s)$$

gdzie

<WŁ>	-	<warunek łączenia>
<warunek łączenia>	-	A_i <operator> B_i
A_i	-	atrybut relacji r
B_i	-	atrybut relacji s
<operator>	-	element zbioru { =, <, >, <=, >=, ...}
x	-	iloczyn kartezjański relacji:

$$r(R) \times s(S) = p(R \square S) = \{ t \mid t = (t[R] \square t[S]) \wedge t[R] \in r \wedge t[S] \in s \}$$

□ - konkatenacja.

W praktycznych zastosowaniach algebry relacji najczęściej używana jest operacja tzw. równołączenia, tzn. w warunku łączenia jest użyty operator = .

3. Pojęcie języków zapytań w relacyjnych bazach danych. Poziomy języków zapytań

W relacyjnej bazie danych można wyróżnić następujące operacje:

- aktualizacja,
- wyszukiwanie,
- przetwarzanie (np. z użyciem operacji arytmetycznych),
- zapamiętywanie i raportowanie.

Język zapytań obejmuje wyszukiwanie oraz zapamiętywanie i raportowanie danych.

Według Kozielskiego [2] można wyróżnić 4 poziomy języków zapytań (tab. 1).

Wraz z poziomem j. zapytań rośnie zwięzłość zapisu.

Przyjęto, że sformułowanie "program na poziomie ... " jest równoważne sformułowaniu "program w języku zapytań na poziomie ..."

4. Realizacja pomiarów czasu rozwiązywania dla wybranych zadań wyszukiwania

Pomiary realizowano w celu uzyskania odpowiedzi na następujące problemy :

1. Jak różnią się czasy rozwiązywania tego samego zadania za pomocą programów w j. zapytań na różnych poziomach dla danego systemu?

2. Jak różnią się dla rozpatrywanych systemów czasy rozwiązywania tego samego zadania za pomocą takiego samego programu i dla tych samych danych?

3. Jak wpływa wzrost złożoności zadania na czas rozwiązywania?

Aby ocenić złożoność zadań przyjęto, że operacją zajmującą najwięcej czasu jest łączenie. Przyjęto, że rozmiarem zadania dla łączenia jest iloczyn liczby krotek obu łączonych relacji.

Pomiary realizowano dla bazy danych PROJEKTY w dwóch wariantach wielkości nazwanych BAZA 1 i BAZA 2.

Sformułowano dwa zadania wyszukiwania.

Tabela 1

Poziomy języków zapytań

Nr	Nazwa	Własności	Typowe instr. dBASE
1	Operowanie na poszczególnych krotkach	Wyszukiwanie krotek (np. o zadanych wartościach atrybutów), najczęściej z użyciem pętli programowych.	SEEK, SKIP - dla indeksowanej bazy danych, LOCATE, CONTINUE - gdy baza nie jest indeksowana.
2	Algebra relacji	Działania na całych relacjach przy użyciu operatorów algebry relacji.	COPY lub SET FILTER - selekcja, projekcja, JOIN, SET RELATION - łączenie, projekcja.
3	Rachunek relacji i rachunek dziedzin	Opis warunków nałożonych na rozwiązanie. Opis zawiera m. in. nazwy atrybutów oraz nazwy relacji, z których te atrybuty pochodzą.	Dla rachunku relacji: instr. SELECT j. SQL, dla rachunku dziedzin: język QBE.
4	Relacja uniwersalna	Opis warunków nałożonych na rozwiązanie. Operuje się jedynie nazwami atrybutów, abstrahując od ich rozmieszczenia w relacjach.	Poziom nie zaimplementowany w badanych systemach.

Tabela 2

Rozmiary relacji tworzących testową bazę danych PROJEKTY

	BAZA 1		BAZA 2	
Relacja	L. krotek	Zajętość [kB]	L. krotek	Zajętość [kB]
MAGAZYN	5 000	130	10 000	260
DCP	1 880	67	3 760	135
BIURO_PR	2 000	42	4 000	84

ZADANIE 1. Podaj numery magazynów z częściami do projektu o numerze 'P1'.

ZADANIE 2. Podaj numery magazynów z częściami do projektów realizowanych w biurze o numerze 'B1'.

Zapis zadania 1 przy użyciu operatorów algebry relacji jest następujący:

$$R1 := \sigma_{NRP='P1'}(DCP)$$

$$WYNIK := \pi_{NRM}(\ MAGAZYN \underset{NRC}{\bowtie} R1\ )$$

Zapis zadania 2 przy użyciu operatorów algebry relacji jest następujący:

$R21 := \sigma_{NRB='B1'}(BIURO\_PR)$

$R22 := DCP \bowtie_{NRP} R21$

$WYNIK := \pi_{NRM}(MAGAZYN \bowtie_{NRC} R22)$

Użyte łączenia są równołączeniami przy warunku równości wartości atrybutu podanego pod znakiem łączenia dla obu łączonych relacji.

Dane w relacjach zostały tak dobrane, aby dla każdego łączenia, w każdej z obu łączonych relacji, warunek łączenia spełniała więcej niż jedna krotka.

Przybliżone rozmiary zadań, liczone jako iloczyn liczby krotek łączonych relacji, przedstawia tabela 3.

Tabela 3

Przybliżone rozmiary zadań użytych do pomiarów

	BAZA 1	BAZA 2
ZADANIE 1	$110*10^5$	$4*10^5$
ZADANIE 2	$13*10^5$	$110*10^5$

Programy rozwiązujące oba zadania zapisano w j. zapytań na trzech pierwszych poziomach. Ze względu na kompatybilność j. zapytań badanych systemów wszystkie systemy badano tymi samymi programami. Wyjątkiem były programy w języku SQL (na poziomie rachunku relacji), ponieważ w badanych systemach występują niewielkie różnice w składni tego języka. Czas rozwiązywania mierzono programem TM.EXE.

W celu zilustrowania wzrostu zwięzłości i nieproceduralności programów przy wzroście poziomu języka zapytań w tabeli 4 zestawiono programy (w j. zapytań na różnych poziomach) rozwiązujące zadanie 1.

Z praktyki programowania wiadomo, że wraz ze wzrostem poziomu języka zapytań rzadko wzrasta szybkość działania programów. W celu zwiększenia szybkości działania programów wyszukujących dane na poziomie algebry relacji Piec [3] zaprojektował i zaimplementował algorytm szybkiego łączenia. Program z tab. 4 na poziomie algebry relacji wykorzystujący szybkie łączenie jest pokazany w przykładzie 1.

Tabela 4

Teksty programów rozwiązujących zadanie 1 na różnych poziomach języka zapytań

OPEROWANIE NA KROTKACH	ALGEBRA RELACJI	RACHUNEK RELACJI

set safety off
set talk off
set exact on
 	
!tm start
sele 1
use MAGAZYN
copy stru to WYNIK fiel NRM
INDEX on NRC to NRC_MAG
sele 2
use WYNIK
sele 3
use DCP
index on NRP to NRP_DCP
!tm stop
 	
!tm start
seek 'P1'
if found ()
do while NRP = 'P1' .and.
.not. eof()
sele MAGAZYN
seek DCP->NRC
if found()
do while NRC = DCP -> NRC
.and. .not. eof()
sele WYNIK
appe blank
replace NRM with
MAGAZYN->NRM
sele MAGAZYN
skip
enddo
endif
sele DCP
skip
enddo
endif
!tm stop
 	
set talk on
set safety on
set exact off
	set safety off
	set talk off
	set exact on
	
	!tm start
	sele 2
	use DCP
	copy to R1 for NRP = 'P1'
	use R1
	sele 1
	use MAGAZYN
	join with R1 to WYNIK
	for NRC = R1->NRC
	field NRM
	!tm stop
	
	set safety on
	set talk on
	set exact off
		set safety off
		set talk off
		set exact on
		
		!tm start
		select MAGAZYN.NRM
		from MAGAZYN, DCP
		where DCP.NRP = 'P1' .and.
		MAGAZYN.NRC = DCP.NRC
		save to temp WYNIK(NRM) keep;
		!tm stop
		
		set safety on
		set talk on
		set exact off

Przykład 1. Program rozwiązujący zadanie 1 przy użyciu szybkiego łączenia

```
set talk off
set safety off
set exact on

!tm start
sele 1
use DCP
copy to R1 for NRP = 'P1'
use R1
index on NRC to RELWRK%1.$$$
use MAGAZYN
index on NRC to RELWRK%2.$$$
use
!tm stop

!tm start
!eqjoin R1.DBF MAGAZYN.DBF WYNIK.DBF RELWRK%1.$$$ RELWRK%2.$$$ * NRM
!tm stop

set safety on
set talk on
set exact off
```

W przykładzie 1 instrukcję JOIN zastąpiono wywołaniem programu szybkiego łączenia o nazwie EQJOIN.

Czas rozwiązywania zadań wyszukiwania, w których zastosowano program szybkiego łączenia, został również przebadany.

Pomiarów dokonano na komputerze Boldline S Series :

- procesor Intel 386SX , 16 MHz ;
- pamięć : 1 M ;
- dysk twardy 40 M , śr. czas dostępu 26.8 ms ;
- system operacyjny : DOS 5.0.

W czasie pomiarów pliki z danymi były zawsze na dysku lokalnym.

5. Wyniki pomiarów i wnioski

Rezultaty badań obrazują tabele 5 i 6.

Tabela 5

Wyniki pomiarów dla zadania 1

Z	B	NB	System	JOIN	iEQJO	EQJOI	LOCAT	iSEEK	SEEK	ixSQL	SQLix	SQL
ZADANIE 1	BAZA 1	1	dBASE III +	01:41	00:39	00:04	03:18	00:42	00:17	n/i	n/i	n/i
		2	dBASE IV 1.0	04:49	n/i	n/i	04:22	00:30	00:07	00:44	00:11	01:16
		3	Clipper 87	01:44	n/i	n/i	02:09	00:20	00:03	n/i	n/i	n/i
		4	Clipper 5.0	04:36	n/i	n/i	04:03	00:18	00:02	n/i	n/i	n/i
		5	FoxBASE + 2.0	01:30	00:16	00:04	01:30	00:18	00:04	n/i	n/i	n/i
		6	FoxPro 1.02	01:14	00:14	00:03	01:10	00:14	00:02	n/i	n/i	n/i
		7	FoxPro 2.0	01:19	n/i	n/i	01:11	00:11	00:04	-	00:11*	n/i
		8	Informix 2.0	n/i	n/i	n/i	n/i	n/i	n/i	01:47	00:10	04:30
	BAZA 2	1	dBASE III +	06:01	01:11	00:05	13:22	01:26	00:56	n/i	n/i	n/i
		2	dBASE IV 1.0	24:46	n/i	n/i	16:21	00:56	00:19	01:11	00:47	02:33
		3	Clipper 87	06:35	n/i	n/i	08:21	00:35	00:04	n/i	n/i	n/i
		4	Clipper 5.0	17:52	n/i	n/i	18:00	00:37	00:05	n/i	n/i	n/i
		5	FoxBASE + 2.0	05:42	00:28	00:04	06:18	00:32	00:11	n/i	n/i	n/i
		6	FoxPro 1.02	04:33	00:23	00:04	04:43	00:26	00:04	n/i	n/i	n/i
		7	FoxPro 2.0	04:52	n/i	n/i	04:27	00:17	00:07	-	00:18*	n/i
		8	Informix 2.0	n/i	n/i	n/i	n/i	n/i	n/i	04:25	00:27	17:36

Objaśnienia do tabel 5 i 6

Dane:

wartości - min:sek

n/i - operacja nie zaimplementowana,

n/m - czas nie zmierzony,

* - czas wyszukiwania razem z autom. indeksacją.

Tabela 6

Wyniki pomiarów dla zadania 2

Z	B	NB	System	JOIN	iEQJO	EQJOI	LOCAT	iSEEK	SEEK	ixSQL	SQLix	SQL
ZADANIE 2	BAZA 1	1	dBASE III +	19:38	00:51	00:13	45:05	00:52	04:55	n/i	n/i	n/i
		2	dBASE IV 1.0	>60:00	n/i	n/i	57:06	00:34	00:56	00:56	00:47	01:28
		3	Clipper 87	21:52	n/i	n/i	29:03	00:24	00:14	n/i	n/i	n/i
		4	Clipper 5.0	błąd	n/i	n/i	>45:00	00:23	00:11	n/i	n/i	n/i
		5	FoxBASE + 2.0	18:00	00:21	00:12	19:00	00:22	00:33	n/i	n/i	n/i
		6	FoxPro 1.02	15:12	00:19	00:13	14:42	00:18	00:11	n/i	n/i	n/i
		7	FoxPro 2.0	15:57	n/i	n/i	14:45	00:13	00:14	-	05:23*	n/i
		8	Informix 2.0	n/i	n/i	n/i	n/i	n/i	n/i	02:19	01:36	59:03
	BAZA 2	3	Clipper 87	n/m	n/i	n/i	n/m	00:45	01:13	n/i	n/i	n/i
		4	Clipper 5.0	n/m	n/i	n/i	n/m	00:47	00:59	n/i	n/i	n/i
		5	FoxBASE + 2.0	n/m	00:44	00:46	n/m	00:40	05:16	n/i	n/i	n/i
		6	FoxPro 1.02	n/m	00:34	00:49	n/m	00:31	00:53	n/i	n/i	n/i
		7	FoxPro 2.0	n/m	n/i	n/i	n/m	00:20	01:06	-	21:14*	n/i
		8	Informix 2.0	n/i	n/i	n/i	n/i	n/i	n/i	05:36	08:58	n/i

Znaczenie nagłówków kolumn :

Z - nr zadania (wg rozdz. 4), którego dotyczą dane,

B - nr wariantu bazy (wg tab. 2), którego dot. dane,

NB - nr systemu bazy danych, wprow. dla celów porządkowych,

System - nazwa systemu bazy danych,

JOIN - czas wyszukiwania w programie na poziomie algebry relacji z użyciem instr. JOIN,

iEQJO - czas indeksowania w programie na poziomie algebry relacji z użyciem programu EQJOIN,

EQJOI - czas łączenia w programie na poziomie algebry relacji z użyciem programu EQJOIN,

LOCAT - czas wyszukiwania w programie na poziomie operowania krotkami, bez indeksacji, z użyciem instr. LOCATE,

iSEEK - czas indeksowania w programie na poziomie operowania krotkami z użyciem instr. SEEK,

SEEK - czas wyszukiwania w programie na poziomie operowania krotkami, z indeksacją, z użyciem instr. SEEK,

ixSQL - czas indeksowania w programie na poziomie rachunku relacji (w j. SQL),

SQLix - czas wyszukiwania w programie na poziomie rachunku relacji, z indeksacją, w j. SQL,

SQL - czas wyszukiwania w programie na poziomie rachunku relacji, bez indeksacji, w j. SQL.

Analizę wyników można przeprowadzić w formie odpowiedzi na problemy przedstawione w rozdz. 4.

Ad 1. Najkrótsze czasy rozwiązywania tego samego zadania w danym systemie osiągano dla indeksowanej bazy danych na poziomie operowania na poszczególnych krotkach, wykorzystując instrukcję SEEK, oraz na poziomie algebry relacji, stosując program szybkiego łączenia EQJOIN.

W systemach dBASE III + i FoxBASE + szybszy był zawsze program EQJOIN. W systemie FoxPro 1.02 zastosowanie zarówno SEEK, jak i EQJOIN dawało zbliżone rezultaty (z nieznaczną przewagą EQJOIN dla większej bazy danych). Z pozostałymi systemami program EQJOIN na razie nie współpracuje.

Dla zadań o małym rozmiarze (zadanie 1 dla obu wariantów bazy) rezultaty porównywalne z rezultatami przy zastosowaniu instrukcji SEEK osiągały programy w j. SQL systemów FoxPro i dBASE IV. Dla zadań o większym rozmiarze (wymagających dwóch łączeń) szybkość działania programów w j. SQL była dużo mniejsza niż programów w innych językach.

Wielokrotnie dłuższe czasy rozwiązywania zadań wyszukiwania osiągnięto, operując na bazie danych bez indeksów, stosując w programach operowanie na pojedynczych krotkach (instrukcja LOCATE) oraz algebrę relacji (instrukcja JOIN). Na przykład, w systemie FoxPro 2.0 dla zadania 2 w bazie 1 (rozmiar danych równy $13*10^5$) czas wykonania programu z zastosowaniem JOIN lub LOCATE jest ok. 30–krotnie dłuższy od czasu wykonania programów z zastosowaniem instrukcji SEEK.

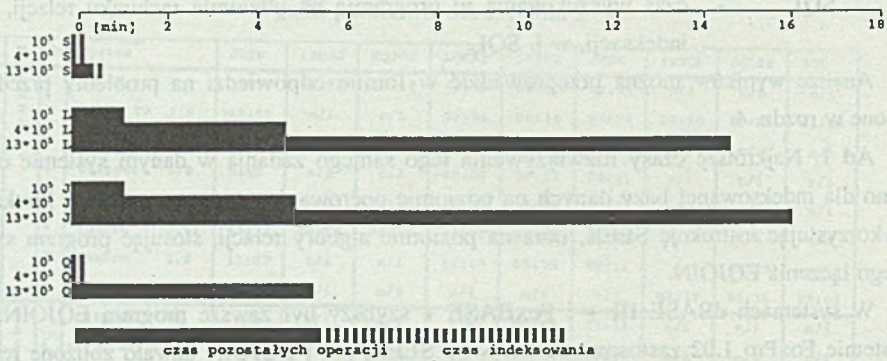
Rys. 2 ilustruje na przykładzie systemu FoxPro 2.0 różnice czasu działania programów na różnych poziomach.

Ad 2. Z odpowiedzi na pytanie 1 wynika, że warto porównywać systemy pod względem konstrukcji językowych dających najkrótsze czasy rozwiązań.

Stosując instrukcję SEEK, porównywalne rezultaty osiągnięto dla systemów Clipper 87, Clipper 5.0, Fox Pro 1.02 i FoxPro 2.0 przy wszystkich rozmiarach zadań.

System FoxBASE + dotrzymywał im kroku dla trzech mniejszych rozmiarów zadań, natomiast dla zadania o największym rozmiarze program wyszukiwania uzyskał w tym systemie czas rozwiązywania ok. 5 razy dłuższy niż te cztery systemy.

Zastosowanie szybkiego łączenia daje podobne rezultaty we wszystkich systemach. Nawet program w systemie dBASE III + dla zadania o największym rozmiarze, przy zastosowaniu programu EQJOIN rozwiązuje to zadanie w czasie 2 min 19 s (w tym czas indeksowania 1 min 36 s). Jest to czas tylko ok. 1.7 raza dłuższy od najkrótszego, osiągniętego w systemie FoxPro 1.02.



S - poziom operow. na krotkach, użycie instr. SEEK,
L - poziom operow. na krotkach, użycie instr. LOCATE,
J - poziom algebry relacji, użycie instr. JOIN,
Q - poziom rachunku relacji, j. SQL
Wartość liczbowa z lewej strony słupków oznacza rozmiar zadania.

Rys. 2. Wykres czasu wykonania programów wyszukiwania w systemie FoxPro 2.0
Fig. 2. The Graph of Query Answering Time in FoxPro 2.0

Użycie j. SQL dawało równoważne rezultaty dla małych rozmiarów zadań we wszystkich systemach.

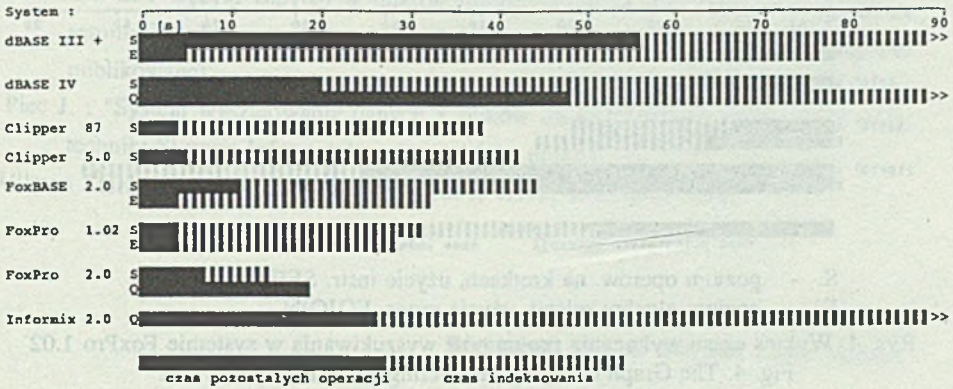
Zadanie o największym rozmiarze najszybciej realizował system INFORMIX, natomiast nie przeprowadzono pomiarów dla systemu dBASEIV (problemy z indeksacją w systemie SQL).

Z pomiarów wynika, że szybkość indeksacji w poszczególnych systemach rośnie wg kolejności : dBASE III +, dBASE IV, Clipper 87, Clipper5.0, FoxBASE +, FoxPro 1.02, FoxPro 2.0.

Warto zauważyć, że nowe wersje systemów (Clipper 5.0, FoxPro2.0) nie wykazują zdecydowanej przewagi pod względem czasu rozwiązywania obu zadań testowych w stosunku do swych wersji poprzednich. Są nawet przypadki wolniejszego działania programów wyszukiwania w nowych wersjach systemów niż w starszych (np. FoxPro dla instrukcji SEEK w zadaniu 2 dla bazy 2).

Rys. 3 ilustruje czas działania programów na różnych poziomach i użytych w różnych systemach. Badano programy o strukturze takiej, jak w tab.4. W programie na poziomie algebry relacji łączenie za pomocą instrukcji JOIN zastąpiono wywołaniem programu EQJOIN.

Ad 3. Wpływ rozmiaru zadania na czas wykonania programu wyszukiwania ilustruje tabela 7.



S - poziom operow. na krotkach, użycie instr. SEEK,
E - poziom algebry relacji, użycie progr. EQJOIN,
Q - poziom rachunku relacji, j. SQL.

Rys. 3. Wykres czasu wykonania programów wyszukiwania dla zadania 1 w bazie 2
Fig. 3. The Graph of Query 1 Answering Time in Database 2

Tabela 7

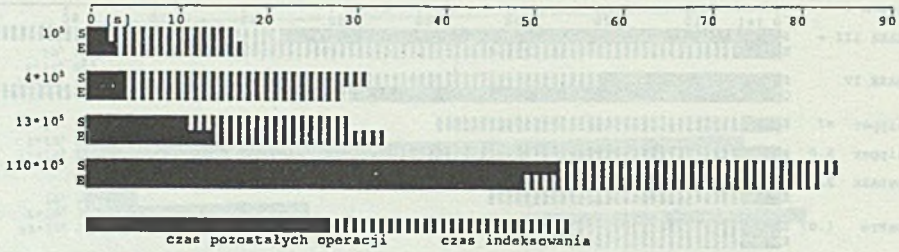
Wpływ rozmiaru zadania na czas jego rozwiązania na przykładzie systemów FoxPro i zadania 1 w bazie 2

	FoxPro 1.02				FoxPro 2.0			
Rozmiar zadania * 100 000	SEEK [s]	LOCATE [min]	EQJOIN [s]	JOIN [min]	SEEK [s]	LOCATE [min]	JOIN [min]	SQL [min]
1	2	1.2	3	1.3	4	1.2	1.3	0.2
4	4	4.8	4	4.5	7	4.5	4.8	0.3
13	11	14.7	13	15.2	11	14.7	15.2	5.4
110	53	n/m	49	n/m	66	n/m	n/m	21.3

n/m - czas nie zmierzony

Dla mniejszych rozmiarów zadań czas rozwiązywania wzrasta mniej więcej liniowo wraz z rozmiarem zadania. Dla większych rozmiarów zadań czas rozwiązania - w przypadku użycia SEEK lub EQJOIN - rośnie wolniej, a w przypadku j. SQL - dużo szybciej od rozmiaru zadania.

Przykładową ilustrację wpływu rozmiaru danych na czas wykonania programów wyszukiwania zawierają rysunki 2 i 4.



S - poziom operow. na krotkach, użycie instr. SEEK,
E - poziom algebry relacji, użycie progr. EQJOIN.

Rys. 4. Wykres czasu wykonania programów wyszukiwania w systemie FoxPro 1.02
Fig. 4. The Graph of Query Answering Time in FoxPro 1.02

6. Podsumowanie

W każdym badanym systemie rozwiązanie zadania wyszukiwania uzyskiwano najszybciej stosując indeksację. W przypadku zastosowania indeksacji czas rozwiązywania zadania wyszukiwania był porównywalny zarówno dla programów z zastosowaniem instrukcji SEEK (poziom operowania na krotkach), jak i dla programów z zastosowaniem programu szybkiego łączenia EQJOIN (poziom algebry relacji).

Wśród badanych systemów najszybsze były FoxPro 2.0, FoxPro 1.02, Clipper 5.0 i Clipper 87.

Wyniki pomiarów umieszczono w tabelach 5 i 6, a omówiono je w rozdziale 5.

Biorąc pod uwagę czas rozwiązania zadania wyszukiwania, Autorzy rekomendują system FoxPro 2.0, ponieważ w tym systemie czasy rozwiązywania większości zadań wyszukiwania były krótsze niż w pozostałych badanych systemach.

Autorzy rekomendują również program szybkiego łączenia EQJOIN, ponieważ programy z zastosowaniem programu EQJOIN są tak samo szybkie, jak programy z zastosowaniem instrukcji SEEK i mają o wiele prostszą strukturę (tabela 4 i przykład 1).

LITERATURA

[1] Kalman D. : "15 Relational Databases : Easy Access, Programming Power" ; PC MAGAZINE, MAY 28, 1991, s. 101-200.

[2] Kozielski S. : "Języki zapytań w mikrokomputerowych bazach danych" ; referat na seminarium Instytutu Informatyki Politechniki Śl., marzec 1992 (materiały nie publikowane).

[3] Piec J. : "System wyszukiwania danych z plików dBASE" ; Zeszyty Naukowe Politechniki Śl. seria Informatyka, zeszyt 16 (w druku).

[4] Ullman J. D. : "Systemy baz danych" ; wyd. I, WNT, Warszawa 1988.

Recenzent: Doc. dr hab. inż. Adam Mrózek

Wpłynęło do Redakcji 10 września 1992 r.

Abstract

This paper contains a coparison of popular relational database management systems. A time of answering a query was assumed as a criterion. Also, an influence of a query language level, and of a size of queries on a time of answering a query was tested.

Database Management Systems which have query languages equivalent to a query language of dBASE and which store data in DBF files was compared.

Additionaly, system INFORMIX SQL was researched.

Tested systems were installed on a separate computer stand.

In every tested system, an answer a query was obtained the most quickly, applied an indexation. In this case, a time of answering a query was comparable to programs with a SEEK instruction (a tuple manipulation level) and with a quick join program EQJOIN (a relational algebra level).

Among researched systems, FoxPro 2.0, FoxPro 1.02, Clipper 5.0, and Clipper 87 were the quickest. All of them realised searching in comparable time, however a speed of an indexaction got smaller in order of series : FoxPro 2.0, FoxPro 1.02, Clipper 5.0, Clipper 87.

Results of measurments are located in the table 5, 6 and they are discussed in the chapter 5.

Taking into consideration a time of answering a query, the Authors recomend the FoxPro 2.0 system, because in this system, a time of answering a majority of queries was shorter than in other comparing systems.

The Authors also recomend the quick join program EQJOIN, because programs with EQJOIN, being as quick as programs with SEEK, have a simpler structure than programs with SEEK (tab. 4 and example 1).