ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1993

Seria: INFORMATYKA z. 22 Nr kol. 1195

Marcin SKOWRONEK

STRUKTURY PLIKÓW DANYCH W RELACYJNYCH BAZACH DANYCH

Streszczenie. W artykule przedstawiono podstawowe, z punktu opracowania procedur dostępu, elementy struktur plików danych stosowanych w bazach danych dBase, FoxBase, Paradox i Informix pracujących pod systemem operacyjnym MS-DOS. Omówione zostały również stosowane w tych bazach danych zasady kodowania danych.

THE STRUCTURE OF DATA FILES IN THE RELATIONAL DATABASES

Summary. In the paper the basic (from the point of view of creating the access procedures) elements of the data files structure used in databases like dBase, FoxBase, Paradox and Informix working under MS-DOS are presented. The coding rules used in these databases are also discussed.

STRUKTUREN DER DATENDATEIEN IN RELATIONALEN DATENBANKSYSTEMEN

Zusammenfassung. In dem Artikel wurden die, im Hinblick auf die Bearbeitung der Zugriffsprozeduren, fundamentalen Strukturelemente der Datendateien vorgestellt, die in den unter dem Betriebssystem MS-DOS laufenden Datenbanksystemen dBase, FoxBase, Paradox und Informix angewandt werden. Es wurden auch die in den obenerwähnten Systemen angewandten Regeln der Datencodierung besprochen.

1. Wprowadzenie

W relacyjnych bazach danych informacje przechowywane są w postaci tabel. Pliki przechowujące tabele nazywane są plikami danych. Wiersze tabeli, stanowiące uporządkowany ciąg danych dotyczących konkretnego obiektu, obserwacji lub zdarzenia, są nazywane również krotkami lub rekordami i są rozróżniane przez numer wiersza. Kolumny tabeli zawierają informacje określonego typu i są nazywane również atrybutami. Kolumnom tabeli przyporządkowane są nazwy. Przecięcie kolumny i wiersza nazywane jest polem. Programy obsługi baz danych dostarczają bogatych narzędzi do:

- określania struktur tabel,
- wprowadzania i aktualizacji danych,
- przetwarzania danych,
- prezentacji danych i wyników przetwarzania.

Spotykane w praktyce zadania przetwarzania często przekraczają możliwości oferowane przez program obsługi bazy danych lub też realizacja przetwarzania za pomocą oferowanych mechanizmów może być zbyt skomplikowana. W wielu przypadkach implementacja algorytmu, za pomocą mechanizmów udostępnianych przez program obsługi bazy danych, może być niecelowa, gdyż istnieją gotowe pakiety procedur realizujących to przetwarzanie napisane w uniwersalnym, ogólnodostępnym języku algorytmicznym. Zakładając, że mechanizmy programu obsługi bazy danych nie umożliwiają włączania procedur napisanych w innych językach programowania lub że napisanie programu przetwarzania będzie skomplikowane, często staniemy przed zadaniem opracowania programu przetwarzania danych, zgromadzonych w plikach bazy danych, w uniwersalnym języku algorytmicznym. W takich przypadkach projektant programu musi również rozwiązać problem dostępu do opisu struktury tabeli i danych umieszczonych w tabelach.

Możliwe są tu dwa rozwiązania. W pierwszym rozwiązaniu można przyjąć, że program będzie odczytywał dane z pośredniego pliku o strukturze niezależnej od struktury plików danych stosowanych w relacyjnej bazie danych. Plik pośredni można zazwyczaj uzyskać z poziomu programu obsługi bazy danych wykorzystując dostępne mechanizmy eksportu danych. Zaletą takiego rozwiązania jest niezależność procedur dostępu od użytej bazy danych. Wadą takiego rozwiązania jest konieczność przechowywania danych zarówno w pliku danych, jak i w pliku pośrednim. Drugie rozwiązaniu może polegać na tym, że projektant opracowuje dla wykorzystywanego systemu bazy danych procedury dostępu bezpośrednio do plików danych. Przyjęcie takiego rozwiązania eliminuje konieczność tworzenia jak i przechowywania danych w pliku pośrednim.

Opracowując procedury dostępu do plików danych projektant musi rozpoznać następujące elementy:

- strukturę plików danych,
- opis struktury rekordu danych,
- strukturę rekordu danych,
- stosowane algorytmy kodowania danych.

Celem artykułu jest omówienie wyżej wymienionych elementów struktury plików danych stosowanych w popularnych bazach danych pracujących pod systemem operacyjnym MS-DOS.

2. Struktura plików typu .DBF

W bazach danych dBase III, dBase III Plus lub FoxBase plikami danych są pliki o rozszerzeniu .DBF [1]. Pliki te zawierają:

- opis struktury tabeli,
- ciąg rekordów stanowiących treść tabeli,
- znacznik końca pliku ($1A).

Opis struktury tabeli składa się z:

- 32-bajtowego nagłówka,
- ciągu 32-bajtowych opisów pól rekordu,
- znacznika końca opisu struktury ($0D).

Istotnymi, dla opracowania procedur dostępu, elementami nagłówka są następujące pola:

Bajty		*Znaczenie*
0	–	identyfikator pliku typu .DBF; przyjmuje wartość $03, gdy rekord nie zawiera pól notatnikowych lub $83, gdy rekord zawiera pola notatnikowe,
4÷7	–	liczba rekordów w tabeli; 4-bajtowa liczba binarna,
8÷9	–	długość opisu struktury; 2-bajtowa liczba binarna,
10÷11	–	długość rekordu; 2-bajtowa liczba binarna.

Liczby binarne w nagłówku zapisywane są w konwencji młodszy bajt, starszy bajt.

W 32-bajtowym opisie pola rekordu istotne dla procedur dostępu są następujące pola:

Bajty		*Znaczenie*
0÷10	–	nazwa pola; ciąg znaków ASCII zakończonych bajtem lub ciągiem bajtów o wartości $00,

11 – kod typu pola; patrz tabela 1,

16 – długość pola; 1-bajtowa liczba binarna,

17 – liczba cyfr po przecinku dla pól numerycznych; 1-bajtowa liczba binarna.

Dopuszczalne typy pól, ich kody oraz zakres długości w bajtach przedstawiono w tabeli 1.

Tabela 1

Kody i zakresy długości pól w plikach typu .DBF

Typ pola	Kod	Zakres długości
znakowy	C	1 ÷ 254
daty	D	8
logiczny	L	1
numeryczny	N	1 ÷ 19
notatnikowy	M	10

Rekord danych stanowi ciąg bajtów, odpowiadających poszczególnym jego polom, poprzedzony jednym bajtem początkowym. Kod bajtu początkowego wykorzystywany jest podczas aktualizacji tabeli i wskazuje programowi obsługi bazy danych, czy dany rekord ma pozostać w tabeli ($20) lub czy ma zostać usunięty ($2A). Rekord danych nie zawiera znaczników końca pól ani znacznika końca rekordu. Zawartości pól rekordu są ciągami znaków ASCII. Wartości nieokreślone w polach znakowych, numerycznych, daty lub notatnikowych zapisywane są jako ciąg spacji. Wartość nieokreśloną w polu logicznym oznacza znak zapytania ('?'). Napis nie wypełniający pola znakowego uzupełniany jest z prawej strony znakami spacji. Napis liczby nie wypełniający pola numerycznego poprzedzany jest znakami spacji z lewej strony. W polu logicznym wartości TRUE i FALSE oznaczane są jedną literą z odpowiednich zbiorów {T, t, Y, y}, {F, f, N, n}. Data zapisywana jest jako ciąg ośmiu cyfr dziesiętnych RRRRMMDD, gdzie R, M, D oznaczają odpowiednio cyfry roku, miesiąca i dnia. Pole notatnikowe zawiera napis liczby dziesiętnej poprzedzonej znakami spacji, oznaczającej numer bloku z pliku o rozszerzeniu .DBT zawierającego informacje z tego pola.

Określenie wartości pola numerycznego wymaga następujących operacji:

- odczytu napisu umieszczonego w polu numerycznym,
- konwersji napisu na wartość liczbową.

Operacja konwersji napisu na wartość liczbową wykonywana jest przez standardowe procedury, na przykład VAL w języku Turbo Pascal lub ATOF w języku Turbo C.

Tabele w plikach typu .DBF mogą zawierać od 1 do 128 kolumn. Wiersz tabeli może zawierać od 1 do 4000 bajtów.

3. Struktura plików typu DB

W bazie danych Paradox (badano wersję 3.5) plikami danych są pliki o rozszerzeniu .DB [2]. Charakterystyczną cechą plików danych w tej bazie jest to, że wiersze tabeli, nazywane dalej rekordami logicznymi, upakowywane są w większe struktury, tzw. rekordy (bloki) fizyczne. Pliki danych zawierają:

- opis struktury tabeli,
- ciąg rekordów fizycznych.

Opis struktury tabeli składa się z:

- 88-bajtowego nagłówka,
- opisu pól,
- ciągu nazw pól.

Istotnymi, dla opracowania procedur dostępu, elementami nagłówka są następujące pola:

Bajty		*Znaczenie*
0÷1	–	długość rekordu logicznego; 2-bajtowa liczba binarna,
2÷3	–	długość opisu struktury; 2-bajtowa liczba binarna,
5	–	długość rekordu fizycznego w kB (1024 B),
6÷9	–	liczba rekordów logicznych; 4-bajtowa liczba binarna,
10÷11	–	liczba użytych rekordów fizycznych; 2-bajtowa liczba binarna,
33	–	liczba kolumn w tabeli (liczba pól w rekordzie); 1-bajtowa liczba binarna.

Liczby binarne w nagłówku zapisywane są w konwencji młodszy bajt, starszy bajt.

Opis pól stanowi ciąg bajtów o długości $6 \cdot M + 83$, gdzie M oznacza liczbę pól w rekordzie. Dla procedur dostępu istotny jest ciąg złożony z $2 \cdot M$ początkowych bajtów. Kolejne pary tego ciągu zawierają dwie 1-bajtowe liczby binarne, oznaczające odpowiednio kod pola i długość pola. Dopuszczalne typy pól, ich kody oraz zakres długości w bajtach przedstawiono w tabeli 2.

Tabela 2

Kody i zakresy długości pól w plikach typu .DB

Typ pola	Kod	Zakres długości
znakowy (A)	1	1 ÷ 255
daty (D)	2	4
całkowity (S)	3	2
waluty ($)	5	8
numeryczny (N)	6	8

Ciąg nazw pól rozpoczyna się od bajtu o numerze $173+6 \cdot M$, gdzie M oznacza liczbę pól w rekordzie. Każda nazwa pola zakończona jest bajtem o wartości zero. Nazwa pola może zawierać do 25 znaków.

Rekord logiczny stanowi ciąg znaków odpowiadających kolejnym jego polom. Rekord logiczny nie zawiera żadnych dodatkowych znaczników. Rekordy logiczne umieszczone są w blokach nazywanych rekordami fizycznymi. Rekord fizyczny zawiera:

- pole informacyjne,
- ciąg rekordów logicznych.

Pole informacyjne tworzą trzy 2-bajtowe liczby binarne, zapisane w konwecji młodszy bajt, starszy bajt, oznaczające:

- numer następnego bloku (następnik),
- numer poprzedniego bloku (poprzednik),
- długość rekordów logicznych poprzedzających ostatni rekord logiczny w bloku (suma).

Pierwszy rekord fizyczny ma przypisany numer 1. Zerowe wartości następnika i poprzednika oznaczają odpowiednio brak następnego lub poprzedniego bloku. Zerowa wartość sumy oznacza, że w rekordzie fizycznym umieszczony jest tylko jeden rekord logiczny.

W zapisie rekordu logicznego stosowane są następujące zasady:

- ciąg bajtów o wartości zerowej wypełniający całe pole oznacza nieokreśloną wartość pola,
- napis w polu znakowym może zawierać dowolne znaki o kodach od 1 do 255; napisy krótsze od długości pola uzupełniane są z prawej strony znakami o wartości zerowej,
- napisy w polach nie będących typu znakowego są kodowane,

- przyjęte zasady kodowania umożliwiają leksykograficzne porównywanie zawartości pól tego samego typu bez konieczności obliczania ich wartości.

Przyjęta zasada kodowania pozwala, w operacjach porównywania pól tego samego rodzaju, traktować zawartość tych pól tak, jak zawartość pól znakowych.

Wartości pól typu daty pamiętane są jako 4-bajtowe liczby binarne, określające liczbę dni według kalendarza Gregoriańskiego od daty 31 grudnia ostatniego roku przed naszą erą. Dacie 1 stycznia roku 1 odpowiada liczba 1.

Pola typu daty (liczba 4-bajtowa) i pola typu całkowitego (liczba 2-bajtowa) zapisywane są w konwencji starszy bajt, młodszy bajt, przy czym stosowane jest kodowanie polegające na negacji bitu znaku. Przykłady kodowania liczb całkowitych przedstawiono w tabeli 3.

Tabela 3

Kodowanie liczb całkowitych w plikach typu .DB

Liczba	513	-720
Obraz pamięci (konwencja młodszy bajt, starszy bajt)		
liczba 2-bajtowa	01 02	30 FD
liczba 4-bajtowa	01 02 00 00	30 FD FF FF
Zapis w pliku danych		
liczba 2-bajtowa	82 01	7D 30
liczba 4-bajtowa	80 00 02 01	7F FF FD 30

Określenie wartości pola typu całkowitego i liczby będącej kodem daty wymaga następujących operacji:

- odczytu ciągu bajtów tworzących pole do bufora roboczego,
- przepisania odczytanych bajtów, ze zmianą kolejności oraz z negacją bitu znaku, do obszaru pamięci zadeklarowanego jako odpowiedni typ całkowity.

Zawartość pola typu numerycznego lub pola typu waluty jest prostą transformacją 8-bajtowego obszaru pamięci, będącego reprezentacją liczby zmiennoprzecinkowej typu DOUBLE (Turbo Pascal, C), przedstawiającej wartość pola. Transformacja ta w zależności od znaku liczby polega na:

a) dla liczb ujemnych
- zanegowaniu wszystkich bitów liczby,
- zapisie liczby w kolejności starszy bajt, młodszy bajt,

b) dla liczb nieujemnych

- zanegowaniu bitu znaku liczby (najstarszy bit najstarszego bajtu),
- zapisie liczby w kolejności starszy bajt, młodszy bajt.

Przykłady kodowania liczb zmiennoprzecinkowych przedstawiono w tabeli 4.

Tabela 4

Kodowanie liczb zmiennoprzecinkowych w plikach typu .DB

Liczba	513	-720
Zapis w pamięci	00 00 00 00 00 08 80 40	00 00 00 00 00 80 86 C0
Zapis w pliku	C0 80 08 00 00 00 00 00	3F 79 7F FF FF FF FF FF

Określenie wartości pola numerycznego lub pola waluty wymaga następujących operacji:

- odczytu ciągu bajtów tworzących pole do bufora roboczego,
- testowania najstarszego bitu pierwszego bajtu w buforze roboczym i dokonania następującej konwersji w zależności od wyniku testowania:
  a) testowany bit = 1; negacja testowanego bitu,
  b) testowany bit = 0; negacja wszystkich bitów dla każdego bajtu bufora roboczego,
- umieszczenia bajtów bufora roboczego, ze zmianą kolejności, w obszarze pamięci zadeklarowanym jako typ DOUBLE (rzeczywisty 8-bajtowy).

Przyjęty sposób kodowania wartości nieokreślonych (zerowa zawartość pola) oznacza, że przyjmują one najmniejszą dopuszczalną dla danego typu wartość. Odczyt takich wartości może powodować błędy przepełnienia. Algorytm konwersji powinien wykrywać takie sytuacje.

Tabele w plikach typu .DB mogą zawierać od 1 do 255 kolumn. Wiersz tabeli może zawierać od 1 do 4000 bajtów dla tabel nieindeksowanych lub od 1 do 1350 bajtów dla tabel indeksowanych.

4. Struktura plików typu .DAT

W bazie danych Informix (badano wersję X.10.06C) tabele przechowywane są w plikach o rozszerzeniu .DAT [3]. W odróżnieniu od baz danych dBase lub Paradox informacje o strukturze tabeli nie są przechowywane w pliku zawierającym treść tabeli. Informacja o nazwach plików bazy danych i ich strukturze przechowywana jest w systemowych

plikach typu .DAT. Każdy plik typu .DAT zawiera ciąg rekordów zakończonych znacznikiem końca rekordu ($0A).

Plik systemowy SYSTABLE.DAT stanowi katalog bazy danych. Rekordy tego pliku opisują pliki typu .DAT (systemowe i użytkownika) tworzące bazę danych. Rekord opisu pliku stanowi ciąg 177 bajtów. Dla procedur dostępu istotne są następujące pola:

Bajty	*Znaczenie*
18÷25	nazwa właściciela pliku; 8-znakowe pole; napis INFORMIX oznacza plik systemowy, napis PCUSER oznacza plik użytkownika,
26÷69	ścieżka dostępu do pliku; 64-znakowe pole,
90÷93	identyfikator pliku; 4-bajtowa liczba binarna,
94÷95	długość rekordu; 2-bajtowa liczba binarna,
96÷97	liczba kolumn; 2-bajtowa liczba binarna,
100÷103	liczba wierszy; 4-bajtowa liczba binarna.

Plik systemowy SYSCOLUM.DAT zawiera opisy kolumn wszystkich tabel (systemowych i użytkownika) zawartych w plikach typu .DAT. Rekord opisu kolumny stanowi ciąg 28 bajtów. Pola rekordu opisu kolumny oznaczają:

Bajty	*Znaczenie*
0÷17	nazwa kolumny; 18-znakowe pole,
18÷21	identyfikator pliku; 4-bajtowa liczba binarna,
22÷23	numer kolumny w tabeli; 2-bajtowa liczba binarna,
24÷25	kod typu kolumny; 2-bajtowe pole,
26÷27	informacje o szerokości kolumny (długości pola); 2-bajtowe pole.

W procedurach dostępu kod typu kolumny może być określony na podstawie bajtu o numerze 25.

Dopuszczalne typy pól, stosowane kody i zakres długości przedstawiono w tabeli 5. Parametr m występujący w oznaczeniu typu dziesiętnego i typu waluty definiuje precyzję liczby (liczbę cyfr znaczących) i może przyjmować wartości od 1 do 32. Zapis $[m/2]$ oznacza część całkowitą z ilorazu $m/2$.

W zapisie rekordu stosowane są następujące zasady:

- pola typu całkowitego, daty i seryjnego są obrazem obszaru pamięci odpowiednich typów całkowitych zapisanych w konwencji starszy bajt, młodszy bajt,
- napisy w polach znakowych, krótsze od długości pola, są uzupełniane z prawej strony znakami spacji,
- pola typu zmiennoprzecinkowego, dziesiętnego i waluty są zapisywane w postaci umożliwiającej leksykograficzne porównywanie zawartości pól tego samego typu bez konieczności obliczania ich wartości,

Tabela 5

Kody i zakresy długości pól w plikach typu .DAT

Typ pola	Kod	Kod długości	Zakres długości
znakowy (Char(n))	0	n	1 ÷ 32767
daty (Date)	7	00 04	4
całkowity (Smallint)	1	00 02	2
całkowity (Integer)	2	00 04	4
seryjny (Serial)	6	00 04	4
zmiennoprzecinkowy (Smallfloat)	5	08 FF	2+8/2
zmiennoprzecinkowy (Float)	5	10 FF	2+16/2
waluty (Money(m,n))	8	m n	2+[m/2]
dziesiętny (Decimal(m,n))	5	m n	2+[m/2]

– zerowa wartość pierwszego bajtu pola znakowego lub najmniejsza dopuszczalna wartość w polach zawierających wartości liczbowe oznacza wartość nieokreśloną.

Wartości pól typu daty pamiętane są jako 4-bajtowe liczby binarne, określające liczbę dni według kalendarza Gregoriańskiego od daty 31 grudnia 1899 roku. Dacie 1 stycznia 1900 roku odpowiada liczba 1.

1 bajt	2 bajt		r-ty bajt	
s	e	a_1	...	a_{r-1}

s – bit znaku,
e – kod cechy,
a_i – cyfry mantysy.

Rys. 1. Format pola typu zmiennoprzecinkowego, dziesiętnego lub waluty
Fig. 1. The format of the floating-point, decimal or money field type

Wartości pól typu zmiennoprzecinkowego, dziesiętnego i waluty są zapisywane w postaci: znak, cecha, mantysa, przy czym dla zapisu mantysy stosowany jest system setkowy. Znaczenie poszczególnych bajtów pola przedstawiono na rys. 1. W systemie setkowym (podstawa p=100) liczba L, przedstawioma w zmiennoprzecinkowym zapisie znormalizowanym, ma postać:

$$L = 100^{w} \cdot M \quad \text{lub} \quad -100^{w} \cdot M; \qquad 0.01 \leq M < 1$$

Niech b_i są cyframi mantysy, natomiast c_i cyframi dopełnienia mantysy

$$M = 0,b_1b_2 \dots b_{r-1}$$
$$1 - M = 0,c_1c_2 \dots c_{r-1}$$

wtedy poszczególne elementy pola zmiennoprzecinkowego określone są następująco:

$$s = \begin{cases} 1, & gdy \quad L \geq 0 \\ 0, & gdy \quad L < 0 \end{cases}$$

$$e = \begin{cases} 64 + w, & gdy \quad L \geq 0 \\ 127 - (64 + w), & gdy \quad L < 0 \end{cases}$$

$$a_i = \begin{cases} b_i, & gdy \quad L \geq 0 \\ c_i, & gdy \quad L < 0 \end{cases}$$

Przykłady kodowania pola zmiennoprzecinkowego przedstawiono w tabeli 6.

Tabela 6

Kodowanie liczb zmiennoprzecinkowych w plikach typu .DAT

Liczba	513	-720
Zapis znormalizowany (p=100)	$100^2 \cdot 0{,}0513$	$-100^2 \cdot 0{,}0720$
Zapis w pliku (5 bajtów)	C2 05 0D 00 00	3D 5C 50 00 00

Określenie wartości pola całkowitego wymaga następujących operacji:

- odczytu ciągu bajtów tworzących pole do bufora roboczego,
- przepisania odczytanych bajtów, ze zmianą kolejności, do obszaru pamięci zadeklarowanego jako odpowiedni typ całkowity.

Ciągi bajtów $80 00 lub $80 00 00 00 oznaczają wartość nieokreśloną, odpowiednio 2- lub 4-bajtowego pola całkowitego.

Określenie wartości pola zmiennoprzecinkowego wymaga następujących trzech operacji:

- odczytu ciągu bajtów tworzących pole do bufora roboczego,
- utworzenia napisu przedstawiającego zapis zmiennoprzecinkowy liczby; przykładowo dla liczby 513 jest to napis 0.0513E4,
- konwersji napisu na wartość liczbową wykorzystując w tym celu standardowe procedury konwersji.

Ciąg bajtów zerowych, wypełniających całe pole zmiennoprzecinkowe, oznacza wartość nieokreśloną.

Tabele w plikach typu .DAT mogą zawierać od 1 do 32767 kolumn. Wiersz tabeli może zawierać od 1 do 32767 bajtów.

5. Podsumowanie

Przedstawione w artykule opisy struktur danych są rezultatem prac wykonywanych podczas opracowania i modyfikacji systemu statystycznej analizy danych – ANALSTAT. Pierwsza wersja systemu ANALSTAT [4] umożliwia wykonywanie zestawu badań statystycznych obejmujących między innymi: obliczanie współczynników korelacji, analizę regresji, testy zgodności rozkładu i analizę wariancji. Dane dla wykonywanych badań odczytywane są bezpośrednio z plików typu .DBF. Opracowywana obecnie nowa wersja tego systemu, oprócz rozszerzenia zakresu oferowanych badań statystycznych, będzie umożliwiała również odczytywanie danych do badań bezpośrednio z plików baz danych Paradox i Informix. Przy opracowywaniu procedur dostępu do plików tych báz danych, najbardziej pracochłonnym zajęciem było rozszyfrowanie struktur plików i stosowanych reguł kodowania, gdyż w dostępnej literaturze brak jest pełnej informacji na ten temat.

Przedstawione w artykule opisy struktur plików danych i zasad kodowania mogą być również użyteczne przy opracowywaniu procedur dostępu do plików baz danych dBase IV oraz FoxPro.

LITERATURA

[1] Using dBase III Plus, Ashton-Tate, 1987.

[2] Paradox V. 3.5, User's Manual, Borland International, 1990.

[3] Informix-SQL, Reference Manual, INFORMIX Software Inc., 1987.

[4] ANALSTAT – System statystycznej analizy danych, Instytut Informatyki Pol. Śląskiej, 1990.

Recenzent: Doc. dr hab. inż. Adam Mrózek

Wpłynęło do Redakcji 1 października 1992 r.

Abstract

The descriptions of the data structures presented in the paper are the result of work under the creation and modification of the ANALSTAT - the system of the data statistical analyses. The first version of the system ANALSTAT [4] offers the means for statistical researches containing amongst others: correlation coefficients computing, regression analyses, distribution testing and variance analyses. The data for computing are read directly from the .DBF type files used in the dBase III plus or FoxBase database. The new version of this system presently being worked out, will also offer data readings directly from the Paradox (the .DB type file) and Informix (the .DAT type file) databases and new statistical tests. The writing of the access procedures to the database files requires knowledge of the file structure, record structure and the coding rules. In the paper the elements mentioned above for the .DBF, .DB and .DAT type files are described as well as examples of the algorithms for the conversion of the numbers fields. The interesting result of the work is the description of the lexicographical coding of the numbers presented in the tables 3 and 4 for the Paradox and in the table 6 for the Informix. The lexicographical format of the number fields make it possible to compare the contents of these fields without having to evaluate their value, which speeds the operation of selection. The description of the data structure files and the coding rules presented in the paper can also be useful for the writing of the access procedures destined for the dBase IV and FoxPro files.