dBASE .DBF File Structure - by Borland Developer Support Staff

Technical Information Database

TI838D.txt dBASE .DBF File Structure
Category :Database Programming
Platform :All
Product :Delphi All

Description:
Sometimes it is necessary to delve into a dBASE table outside the control of the Borland Database Engine (BDE). For instance, if the .DBT file (that contains memo data) for a given table is irretrievably lost, the file will not be usable because the byte in the file header indicates that there should be a corresponding memo file. This necessitates toggling this byte to indicate no such accompanying memo file. Or, you may just want to write your own data access routine.

Below are the file structures for dBASE table files. Represented are the file structures as used for various versions of dBASE: dBASE III PLUS 1.1, dBASE IV 2.0, dBASE 5.0 for DOS, and dBASE 5.0 for Windows.

****
The data file header structure for dBASE III PLUS table file.
****

The table file header:
==========

Byte	Contents	Description
0	1 byte	Valid dBASE III PLUS table file (03h without a memo (.DBT file; 83h with a memo).
1-3	3 bytes	Date of last update; in YYMMDD format.
4-7	32-bit number	Number of records in the table.
8-9	16-bit number	Number of bytes in the header.
10-11	16-bit number	Number of bytes in the record.
12-14	3 bytes	Reserved bytes.
15-27	13 bytes	Reserved for dBASE III PLUS on a LAN.
28-31	4 bytes	Reserved bytes.
32-n	32 bytes each	Field descriptor array (the structure of this array is shown below)
n+1	1 byte	0Dh stored as the field terminator.

n above is the last byte in the field descriptor array. The size of the array depends on the number of fields in the table file.

Table Field Descriptor Bytes
==========

Byte	Contents	Description
0-10	11 bytes	Field name in ASCII (zero-filled).
11	1 byte	Field type in ASCII (C, D, L, M, or N).
12-15	4 bytes	Field data address (address is set in memory; not useful on disk).

16	1 byte	Field length in binary.
17	1 byte	Field decimal count in binary.
18-19	2 bytes	Reserved for dBASE III PLUS on a LAN.
20	1 byte	Work area ID.
21-22	2 bytes	Reserved for dBASE III PLUS on a LAN.
23	1 byte	SET FIELDS flag.
24-31	1 byte	Reserved bytes.

Table Records
==========

The records follow the header in the table file. Data records are preceded by one byte, that is, a space (20h) if the record is not deleted, an asterisk (2Ah) if the record is deleted. Fields are packed into records without field separators orrecord terminators. The end of the file is marked by a single byte, with the end-of-file marker, an OEM code page character value of 26 (1Ah). You can input OEM code page data as indicated below.

Allowable Input for dBASE Data Types
==========

Data Type	Data Input
C (Character)	All OEM code page characters.
D (Date)	Numbers and a character to separate month, day, and year (stored internally as 8 digits in YYYYMMDD format).
N (Numeric)	- . 0 1 2 3 4 5 6 7 8 9
L (Logical)	? Y y N n T t F f (? when not initialized).
M (Memo)	All OEM code page characters (stored internally as 10 digits representing a .DBT block number).

Binary, Memo, and OLE Fields And .DBT Files
==========

Memo fields store data in .DBT files consisting of blocks numbered sequentially (0, 1, 2, and so on). The size of these blocks are internally set to 512 bytes. The first block in the .DBT file, block 0, is the .DBT file header.

Memo field of each record in the .DBF file contains the number of the block (in OEM code page values) where the field's data actually begins. If a field contains no data, the .DBF file contains blanks (20h) rather than a number.

When data is changed in a field, the block numbers may also change and the number in the .DBF may be changed to reflect the new location.

This information is from the Using dBASE III PLUS manual, Appendix C.

****
The data file header structure for dBASE IV 2.0 table file.
****

File Structure:
==========

Byte	Contents	Meaning
0	1byte	Valid dBASE IV file; bits 0-2 indicate version

		number, bit 3 the presence of a dBASE IV memo file, bits 4-6 the presence of an SQL table, bit 7 the presence of any memo file (either dBASE III PLUS or dBASE IV).
1-3	3 bytes	Date of last update; formattted as YYMMDD.
4-7	32-bit number	Number of records in the file.
8-9	16-bit number	Number of bytes in the header.
10-11	16-bit number	Number of bytes in the record.
12-13	2 bytes	Reserved; fill with 0.
14	1 byte	Flag indicating incomplete transaction.
15	1 byte	Encryption flag.
16-27	12 bytes	Reserved for dBASE IV in a multi-user environment.
28	1 bytes	Production MDX file flag; 01H if there is an MDX, 00H if not.
29	1 byte	Language driver ID.
30-31	2 bytes	Reserved; fill with 0.
32-n*	32 bytes each	Field descriptor array (see below).
n + 1	1 byte	0DH as the field terminator.

* n is the last byte in the field descriptor array. The size of the array depends on the number of fields in the database file.

The field descriptor array:
==========

Byte	Contents	Meaning
0-10	11 bytes	Field name in ASCII (zero-filled).
11	1 byte	Field type in ASCII (C, D, F, L, M, or N).
12-15	4 bytes	Reserved.
16	1 byte	Field length in binary.
17	1 byte	Field decimal count in binary.
18-19	2 bytes	Reserved.
20	1 byte	Work area ID.
21-30	10 bytes	Reserved.
31	1 byte	Production MDX field flag; 01H if field has an index tag in the production MDX file, 00H if not.

Database records:
==========

The records follow the header in the database file. Data records are preceded by one byte; that is, a space (20H) if the record is not deleted, an asterisk (2AH) if the record is deleted. Fields are packed into records without field separators or record terminators. The end of the file is marked by a single byte, with the end-of-file marker an ASCII 26 (1AH) character.

Allowable Input for dBASE Data Types:
==========

Data	Type	Data Input
C	(Character)	All OEM code page characters.
D	(Date)	Numbers and a character to separate month, day, and year (stored internally as 8 digits in YYYYMMDD format).
F	(Floating point binary numeric)	- . 0 1 2 3 4 5 6 7 8 9

N	(Binary coded decimal numeric)	- . 0 1 2 3 4 5 6 7 8 9
L	(Logical)	? Y y N n T t F f (? when not initialized).
M	(Memo)	All OEM code page characters (stored internally as 10 digits representing a .DBT block number).

Memo Fields And .DBT Files

Memo fields store data in .DBT files consisting of blocks numbered sequentially (0, 1, 2, and so on). SET BLOCKSIZE determines the size of each block. The first block in the .DBT file, block 0, is the .DBT file header.

Each memo field of each record in the .DBF file contains the number of the block (in OEM code page values) where the field's data actually begins. If a field contains no data, the .DBF file contains blanks (20h) rather than a number.

When data is changed in a field, the block numbers may also change and the number in the .DBF may be changed to reflect the new location.

This information is from the dBASE IV Language Reference manual, Appendix D.

****

The data file header structure for dBASE 5.0 for DOS table file.

****

The table file header:

Byte	Contents	Description
0	1 byte	Valid dBASE for Windows table file; bits 0-2 indicate version number; bit 3 indicates presence of a dBASE IV or dBASE for Windows memo file; bits 4-6 indicate the presence of a dBASE IV SQL table; bit 7 indicates the presence of any .DBT memo file (either a dBASE III PLUS type or a dBASE IV or dBASE for Windows memo file).
1-3	3 bytes	Date of last update; in YYMMDD format.
4-7	32-bit number	Number of records in the table.
8-9	16-bit number	Number of bytes in the header.
10-11	16-bit number	Number of bytes in the record.
12-13	2 bytes	Reserved; filled with zeros.
14	1 byte	Flag indicating incomplete dBASE transaction.
15	1 byte	Encryption flag.
16-27	12 bytes	Reserved for multi-user processing.
28	1 byte	Production MDX flag; 01h stored in this byte if a production .MDX file exists for this table; 00h if no .MDX file exists.
29	1 byte	Language driver ID.
30-31	2 bytes	Reserved; filled with zeros.
32-n	32 bytes each	Field descriptor array (the structure of this array is shown below)
n+1	1 byte	0Dh stored as the field terminator.

n above is the last byte in the field descriptor array. The size of the array depends on the number of fields in the table file.

Table Field Descriptor Bytes
==========

Byte	Contents	Description
0-10	11 bytes	Field name in ASCII (zero-filled).
11	1 byte	Field type in ASCII (B, C, D, F, G, L, M, or N).
12-15	4 bytes	Reserved.
16	1 byte	Field length in binary.
17	1 byte	Field decimal count in binary.
18-19	2 bytes	Reserved.
20	1 byte	Work area ID.
21-30	10 bytes	Reserved.
31	1 byte	Production .MDX field flag; 01h if field has an index tag in the production .MDX file; 00h if the field is not indexed.

Table Records
==========

The records follow the header in the table file. Data records are preceded by one byte, that is, a space (20h) if the record is not deleted, an asterisk (2Ah) if the record is deleted. Fields are packed into records without field separators orrecord terminators. The end of the file is marked by a single byte, with the end-of-file marker, an OEM code page character value of 26 (1Ah). You can input OEM code page data as indicated below.

Allowable Input for dBASE Data Types
==========

Data Type	Data Input
C (Character)	All OEM code page characters.
D (Date)	Numbers and a character to separate month, day, and year (stored internally as 8 digits in YYYYMMDD format).
F (Floating point binary numeric)	- . 0 1 2 3 4 5 6 7 8 9
N (Numeric)	- . 0 1 2 3 4 5 6 7 8 9
L (Logical)	? Y y N n T t F f (? when not initialized).
M (Memo)	All OEM code page characters (stored internally as 10 digits representing a .DBT block number).

Memo Fields And .DBT Files
==========

Memo fields store data in .DBT files consisting of blocks numbered sequentially (0, 1, 2, and so on). SET BLOCKSIZE determines the size of each block. The first block in the .DBT file, block 0, is the .DBT file header.

Each memo field of each record in the .DBF file contains the number of the block (in OEM code page values) where the field's data actually begins. If a field contains no data, the .DBF file contains blanks (20h) rather than a number.

When data is changed in a field, the block numbers may also change and the number in the .DBF may be changed to reflect the new location.

Unlike dBASE III PLUS, if you delete text in a memo field, dBASE 5.0 for DOS may reuse the space from the deleted text when you input new text. dBASE III PLUS always appends new text to the end of the .DBT file. In dBASE III PLUS, the .DBT file size grows whenever new text is added, even if other text in the file is deleted.

This information is from the dBASE for DOS Language Reference manual, Appendix C.

****
The data file header structure for dBASE 5.0 for Windows table file.
****

The table file header:
==========

Byte	Contents	Description
0	1 byte	Valid dBASE for Windows table file; bits 0-2 indicate version number; bit 3 indicates presence of a dBASE IV or dBASE for Windows memo file; bits 4-6 indicate the presence of a dBASE IV SQL table; bit 7 indicates the presence of any .DBT memo file (either a dBASE III PLUS type or a dBASE IV or dBASE for Windows memo file).
1-3	3 bytes	Date of last update; in YYMMDD format.
4-7	32-bit number	Number of records in the table.
8-9	16-bit number	Number of bytes in the header.
10-11	16-bit number	Number of bytes in the record.
12-13	2 bytes	Reserved; filled with zeros.
14	1 byte	Flag indicating incomplete dBASE IV transaction.
15	1 byte	dBASE IV encryption flag.
16-27	12 bytes	Reserved for multi-user processing.
28	1 byte	Production MDX flag; 01h stored in this byte if a production .MDX file exists for this table; 00h if no .MDX file exists.
29	1 byte	Language driver ID.
30-31	2 bytes	Reserved; filled with zeros.
32-n	32 bytes each	Field descriptor array (the structure of this array is shown below)
n+1	1 byte	0Dh stored as the field terminator.

n above is the last byte in the field descriptor array. The size of the array depends on the number of fields in the table file.

Table Field Descriptor Bytes
==========

Byte	Contents	Description
0-10	11 bytes	Field name in ASCII (zero-filled).
11	1 byte	Field type in ASCII (B, C, D, F, G, L, M, or N).
12-15	4 bytes	Reserved.
16	1 byte	Field length in binary.

17	1 byte	Field decimal count in binary.
18-19	2 bytes	Reserved.
20	1 byte	Work area ID.
21-30	10 bytes	Reserved.
31	1 byte	Production .MDX field flag; 01h if field has an index tag in the production .MDX file; 00h if the field is not indexed.

Table Records

The records follow the header in the table file. Data records are preceded by one byte, that is, a space (20h) if the record is not deleted, an asterisk (2Ah) if the record is deleted. Fields are packed into records without field separators orrecord terminators. The end of the file is marked by a single byte, with the end-of-file marker, an OEM code page character value of 26 (1Ah). You can input OEM code page data as indicated below.

Allowable Input for dBASE Data Types

Data Type	Data Input
B (Binary)	All OEM code page characters (stored internally as 10 digits representing a .DBT block number).
C (Character)	All OEM code page characters.
D (Date)	Numbers and a character to separate month, day, and year (stored internally as 8 digits in YYYYMMDD format).
G (General	All OEM code page characters (stored internally as 10 digits or OLE) representing a .DBT block number).
N (Numeric)	- . 0 1 2 3 4 5 6 7 8 9
L (Logical)	? Y y N n T t F f (? when not initialized).
M (Memo)	All OEM code page characters (stored internally as 10 digits representing a .DBT block number).

Binary, Memo, and OLE Fields And .DBT Files

Binary, memo, and OLE fields store data in .DBT files consisting of blocks numbered sequentially (0, 1, 2, and so on). SET BLOCKSIZE determines the size of each block. The first block in the .DBT file, block 0, is the .DBT file header.

Each binary, memo, or OLE field of each record in the .DBF file contains the number of the block (in OEM code page values) where the field's data actually begins. If a field contains no data, the .DBF file contains blanks (20h) rather than a number.

When data is changed in a field, the block numbers may also change and the number in the .DBF may be changed to reflect the new location.

Unlike dBASE III PLUS, if you delete text in a memo field (or binary and OLE fields), dBASE for Windows (unlike dBASE IV) may reuse the space from the deleted text when you input new text. dBASE III PLUS always appends new text to the end of the .DBT file. In dBASE III PLUS, the .DBT file size grows whenever new text is added, even if other text in the file is deleted.

This information is from the dBASE for Windows Language Reference manual,

Appendix C.

Reference:

7/16/98 4:33:55 PM