Help Index

DataWorks Concepts
- Data Dictionary
- Quick Keys
- Defined Terms
- Expressions, Functions, and Operators
- Error Messages

Procedures
- Windows Keys
- System Menu

File Directory Commands
- File Menu
- Edit Menu
- Tools Menu
- Settings Menu

File Structure Commands
- Edit Structure
- Print Structure

Data Table Commands
- Edit Menu
- Query Menu
- Print Data
- Filter Data
- Join Command

Memo Commands
- File Menu
- Edit Menu

The Index contains a list of all Help topics available.

For information on how to use Help, press F1 or choose Using Help from the Help menu.

Data Dictionary
DataWorks manages relational database files structured like dBase III or dBase III+ files. The generic term for these files is xBase. A data file usually has one or more index files associated with it which provide almost immediate access to any record. The index keys also provide the links required to establish file relationships.

DataWorks revolves around the concept of a data dictionary. The data dictionary (which is a collection of files) describes a set of xBase files in any given directory. A complete set of DataWorks dictionary files consists of:

dwfiles.dbf describes each .dbf in the directory as to physical file name and a user-entered description of the file's contents. The DataWorks main window is a tabular display of the contents of this file.

dwfields.dbf describes each field in the defined file. The field name, data type (character, numeric, logical, date, or memo), the field width and number of decimals (for numeric fields), as well as a description of the contents of the field resides in this file.

dwindex.dbf identifies each .ntx file associated with the defined .dbf. The indexing expression is also kept in this file.

dwforms.dbf describes the layout of a data entry dialog box. Generic forms are built directly from the dwfields file. The user may define custom forms with the TOOLS FORMS command.

All DataWorks dw files must be present in the user directory to allow the definition and modification of existing xBase files to DataWorks, or the creation of new xBase files. A basic set of files may be copied from the distribution diskette into a user database directory.

Quick Keys

Any file accessed with an index may have its current record pointer (i.e., the record currently shown in reverse video) positioned by typing in the the characters that make up the index key. For example, from the main file directory window, typing the letter "D" will position the pointer to the first file name that starts with a "D". Typing in the letter "X" would then further refine the search to the file that began with the letters "DX", and so on. The entered quick key is displayed (temporarily) in the clock window at the bottom right of the main window.

System Menu
Every DataWorks window has a system menu accessed by pointing at the bar in the upper left corner of the window. Every DataWorks display window is closed using the system menu. If the main file directory window is closed, DataWorks terminates.

The system menu may also be accessed through the keyboard with ALT-SPACEBAR. A window may be closed with ALT_F4.

Expressions, Functions, and Operators
Complete definitions of expressions, functions, and operators may be accessed by choosing one of the topics below:

Expressions
Constants
Operators
Operator Precedence
Functions

Expressions
Expressions are character strings that consist of field names, functions, constants, and operators. For example, index keys are created by supplying an expression to DataWorks.

An expression may be as simple as a single field name (e.g., cust_name) or as complicated as an IIF function which returns complex expressions (e.g., IFF(left(phone_num,1)=" ","No phone on file",area_code+phone_num)).

The IIF example expressed in normal language would read as "If the first character of the phone_num field is blank, output the phrase 'No phone on file'; otherwise, output the area code plus the phone number". This expression contains two functions (IIF() and LEFT()), two constants (a space between the two quotation marks and the phrase "No phone on file"), two field names (phone_num and area_code), and two operators (the relational operator equal sign = and the string concatenation operator plus sign +).

Expressions are used in index keys, filter definitions, definition of beginning of file and end of file logic to a user table, in statements used to join (or relate) one file to another, and in statements used to define the contents of a display column when defining a table or a report row.

All expressions return a value of a specific type - either character, numeric, date, or logical. In many cases, Dataworks requires that an expression return a value of a specific type. For example, when defining a filter expression to limit the viewable records, the expression must evaluate as logical (i.e., either TRUE or FALSE). A conditional filter may be defined that limits a view to all customer records that begin with the letter "A". This condition could be expressed as LEFT(cust_name,1)="A". Dataworks would interpret this as "If the leftmost character of the field CUST_NAME is an "A", then display the record". The presence of a relational operator (in this case, the equal sign) generally denotes an expression that will evaluate as logical.

Expressions may be entered in upper or lower case.

Constants
An expression may contain one or more numeric, character, or logical constants. An expression which consists of a single constant is not very useful. Constants are usually used within more complex expressions.

A numeric constant represents a number. For example, 4, 9.21, and -26 are all numeric constants.

Character or *string* constants are always delimited with quotation marks, either single or double. "This is a string", 'so is this', and "John has 3 apples" are all character constants. A string that contains either a double or single quotation mark must be delimited with the other mark. For example, "John's apple" is a valid string. 'John's apple' is not a valid string.

Logical constants are represented by .TRUE. or .FALSE.. Note the leading and trailing periods. .T. and .F. are valid abbreviations for the logical constants and the letters must be bounded by periods on both sides.

Operators
Operators are signs used to manipulate fields, constants, and the results of functions. A plus sign (+) is used as an *Add Operator* in the expression 4+5. Two numeric constants are added together to return the numeric value 9.

Operators are *type specific*. For example, arithmetic operators must act on numeric types. The Divide Operator (/) only acts on numeric types. Some operators perform double duty. The Plus and Minus signs are both arithmetic and string operators. DataWorks determines the appropriate operation according to the type of data being acted upon. The data types on either side of a relational operator must be the same (i.e., strings must be compared to strings and numbers must be compared to numbers). Functions which change the data type may be used to convert operands for use in relational expressions.

The only *mixed operands* allowed are involved in *Date Arithmetic*. A numeric constant , field, or expression may be added to or subtracted from a date type. Dates subtracted from dates yield a numeric type (i.e., the number of days between two dates).

For more information on operators, select one of the following topics:

Numeric Operators
Relational Operators
Logical Operators
Character (String) Operators

Numeric Operators

+	Addition
-	Subtraction
*	Multiplication
/	Division
^ or **	Exponentiation
()	Groups sets of numbers (evaluation order)

Relational Operators

=	Equal to
#	Not equal to
<>	Not equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
$	Is contained in the set or is a subset of

All relational operators return a Logical result. All operators except the Contains($) operator work on numeric, character, or date values. The $ operator works on two values of type character and returns true if the first value is contained in the second (e.g., "DC"$"ABDC" returns .TRUE.).

Logical Operators

.AND.	both expressions are true
.OR.	either expression is true
.NOT.	either expression is false

Note the leading and trailing periods that delimit a logical operator.

Character (String) Operators

+ Concatenates (joins) two or more character expressions. Trailing blank spaces in the expressions will be placed at the end of each expression.

- Concatenates two or more expressions. Trailing blank spaces will be removed from the expression preceding the operator and placed at the end of the expression following the minus sign operator.

Operator Precedence

When more than one type of operator appears in an expression, the order of evaluation is as follows:

string
numeric
relational
logical

Expressions containing more than one operator are evaluated from left to right. Parentheses can be used to change the precedence level of operators (see example below). If parentheses are nested, the innermost set is evaluated first.

Numeric operators are evaluated as follows:

operators contained in parentheses
exponentiation
multiplication and division
addition and subtraction

Evaluation order may be altered with parentheses:

1+2*3+4 = 11
(1+2)*3+4 = 13
(1+2)*(3+4) = 21

Functions
Functions may be used as expressions or parts of expressions. Functions always return a value.

One of the most common uses of functions is to convert one data type into another. Functions can also extract system and database-specific information.

Functions are formatted as *FunctionName(Parameters)*. The number and type of parameters contained within the function parentheses depend on the specific function being called.

The following functions are available. For more information, select the specific command.

Function	**Returns**
CTOD(Char_Value)	Character to date
DATE()	System date
DAY(Date_Value)	Numeric day
DEL()	"*" if deleted
DELETED()	.TRUE. if deleted
DESCEND()	create descending index
DTOC(Date_Value)	Date to character
DTOS(Date_Value)	Date to string
IIF(Logical, True Result, False Result)	
LEFT(Char_Value, Length)	Leftmost *n* characters
LEN(Char_Value)	Length of character expression
MONTH(Date_Value)	Numeric month
RECCOUNT()	Number of records in the file
RECNO()	Record number
RIGHT(Char_Value, Length)	Rightmost *n* characters
SOUNDEX(Char_Value)	String to phonetic complement
SPACE(Length)	Generate spaces for specified length
STOD(Char_Value)	String to date
STR(Number, Len, Dec)	Numeric value to string
SUBSTR(Char_Value, Start, Length)	Substring
TIME()	System time as string
TRIM(Char_Value)	Remove trailing spaces
UPPER(Char_Value)	Convert to uppercase
VAL(Char_Value)	Character to numeric value
YEAR(Date_Value)	Numeric Year

CTOD(Char_Value)

Character to date function.
Converts a character value in the form "MM/DD/YY" into a date value.
Example: CTOD("07/22/91") returns a date in the form CCYYMMDD 19910722

DATE()

System date function.
Returns the system date as a date value.
Example: DTOC(DATE()) returns "07/22/91" If the date is July 22, 1991

DAY(Date_Value)

Numeric day function.
Returns the day in a date_value as a number.
Example: DAY(DATE()) returns 22 if the date is July 22, 1991

DEL()

Deletion flag function.
Returns "*" if the current record has been flagged for deletion.
Returns a character space if not deleted.
Example: IIF(DEL() = "*", "Deleted", "Not Deleted")

DELETED()

Logical delete function.
Returns .TRUE. if the current record has been flagged for deletion.
Example: IIF(DELETED(), "Deleted", "Not Deleted")

DESCEND()

Create descending index.

An entire index expression or an element of a complex index expression may be encapsulated within a DESCEND function to reverse the normal ascending sequence.

Example: UPPER(cust_name) + DESCEND(DTOS(order_day))

DTOC(Date_Value)

Date to character function.
Converts a date value into a character string in the format "MM/DD/YY".
Example: DTOC(DATE()) returns "07/22/91" if the date is July 22, 1991

DTOS(Date_Value)

Date to string function.

Converts a date value into a character string in the format "CCYYMMDD".

Should always be used in index expressions if a date field is part of the index key expression.

Example: DTOS(DATE()) returns "19910722" if the date is July 22, 1991

IIF(Logical_Value, True_Result, False_Result)

Logical if function.

If Logical_Value is evaluated as .TRUE., then the expression represented by True_Result is returned; otherwise, the expression represented by False_Result is returned.

True_Result and False_Result must be of the same type.

Example 1: IIF(YEAR(DATE()) < 1991, "Last Year", "This Year")

Example 2: IIF(amt_owing>0, amt_owing, 0)

LEFT(Char_Value, Length)

Leftmost characters function.

Returns the characters on the left side of the string for the specified length.

Example: IIF(LEFT(NAME,1)<>"A", "Does not begin with A", "begins with A")

LEN(Char_Value)

Get the length of a character expression.
The length of a TRIMmed blank character expression will be zero.
Example: IIF(LEN(TRIM(NAME))>0, NAME,COMPANY)

MONTH(Date_Value)

Numeric month function.
Returns the month in a date_value as a number.
Example: MONTH(DATE()) returns 7 if the date is July 22, 1991

RECCOUNT()

Record count function.
Returns the number of records in the database.
Example: IIF(RECNO()=RECCOUNT(), "Last record", "Not last record")

RECNO()

Record number function.

Returns the physical record number of the current record. The record's logical position according to the current index is probably not the same as this number. The record number normally reflects the sequence in which the record was entered.

Example: IIF(RECNO()=RECCOUNT(), "Last record", "Not last record")

RIGHT(Char_Value, Length)

Rightmost characters function.
Returns the characters on the right side of the string for the specified length.
Example: RIGHT("ABCDEF", 3) returns "DEF"

SOUNDEX(Char_Value)

Character string to phonetic complement function. Useful for indexing and searching.

Returns a character string in the form AA1111.

Used primarily for indexes on names and descriptions to conserve index file space and simplify lookups where the precise spelling of an item (other than the first two characters) is unknown. Always results in a table display that *approximates* alphabetical order.

Example: SOUNDEX(cust_name) returns a 6 character string

SPACE(Length)

Generate a string containing all spaces for the specified length.

Used with TRIM() to create fixed length concatenations of character fields for indexing or browse column displays.

Example: SUBSTR((TRIM(LASTNAME)+", "+TRIM(FIRSTNAME)+SPACE(40)),1,40)

STOD(Char_Value)

Converts a string representation of a date (in "CCYYMMDD format") to a date_value.
Returns a date_value.
Example: STOD("19910722") returns a date value in date format.

STR(Number, Len, Dec)

Numeric to string function.

Converts a number to a string representation of that number. *Len* is the number of characters in the new string, and *Dec* is the number of decimals.

DataWorks Tools Tables function requires the definition of all columns to evaluate to strings in order to properly calculate their display width. Always define numeric values with the STR() function.

Example: STR(CURRENT+PAST_DUE,9,2) would result in "123456.78" if the sum of the fields CURRENT and PAST_DUE was equal to the number 123,456.78.

If the resulting number is too large for the allotted space, the string is filled with asterisks.

SUBSTR(Char_Value, Start, Length)

Substring function.

Returns a substring of the string represented by Char_Value.

Example: SUBSTR("abcdef,4,3") returns "def" (i.e., extract a substring from "abcdef" beginning with the fourth character for a length of 3)

TIME()

Time of day function.
Returns the system time as a character string in the form HH:MM:SS.
Example 1: TIME() returns 12:00:00 at noon
Example 2: TIME() returns 13:45:00 at one forty-five p.m.

TRIM(Char_Value)

Trim trailing (rightmost) spaces from a character field.

May be used to logically concatenate fields in a browse column or in an index expression AS LONG AS the length of the resultant expression is FIXED.

Example: SUBSTR((TRIM(LASTNAME)+", "+TRIM(FIRSTNAME)+SPACE(40)),1,40)

UPPER(Char_Value)

Convert string to uppercase.

Only alphabetic characters are affected.

Mostly used in index expressions to ensure correct collating sequence for character strings without regard to data entry formats.

Example: UPPER("abCD123g") returns "ABCD123G"

VAL(Char_Value)

String to numeric conversion.

Evaluation is terminated when a second decimal point, the first non-numeric character, or the end of the string is reached.

Example 1: VAL("23") returns 23
Example 2: VAL("12A12") returns 12
Example 3: VAL("-76.5") returns -76.5
Example 4: VAL(" 12.12") returns 12.12
Example 5: VAL("12. 12") returns 12.00
Example 6: VAL("A12") returns 0

YEAR(Date_Value)

Numeric year function.
Returns the year in a date_value as a number.
Example: YEAR(DATE()) returns 1991 if the date is July 22, 1991

File Directory File Menu
The File menu includes commands that enable you to open xBase files, examine and modify their structure, create new files, delete files, import existing files into the DataWorks dictionary, display statistics about a highlighted file, and print a listing of the xBase files in the selected directory.

For more information, select the File menu command name.

Open
Structure
About
New
Delete
Import dbf
Print

File Open Command
When this item is selected, the file name currently displayed in inverse video is opened and a data display window of that file is presented as a table. Each row represents one record and each column represents one field. If user defined display tables built with the Tools Tables command exist for the file, the user may select a custom display format.

A file may be opened by using the arrow keys or the mouse to position the inverse video bar over the desired file name and selecting this command from the FILE MENU. Pressing the Enter key or double clicking the left mouse button will also open the file.

Multiple display windows may be opened by re-selecting the file directory window and picking another file.

If there is one or more index files associated with the requested dbf, a dialog box is presented allowing you to choose the display sequence.

File Structure Command

Opens a file structure window and displays the record layout of the data in the file that has the highlight. By selecting the EDIT menu item from the File Structure window, the structure of the database may be modified.

File About Command
Displays relevant statistics about the file that is currently marked in inverse video.

File New Command

Allows the creation of a new database file. Through a series of dialog boxes, the user assigns a physical file name and description to the file, describes the fields that make up each record in the database, and also defines one or more index files. Records may be added to a new file by opening the file and answering YES to the *Add Records?* prompt.

File Delete Command

The user may delete DataWorks dictionary references to a file and/or the actual physical file and/or its related index files. A dialog box is opened that features a series of check boxes next to each type of dictionary entry or actual file. An "X" in the checkbox means that the item will be deleted.

File Import dbf Command

Existing dbf files in the current database directory that have not been defined to DataWorks may be accessed by using this command. All dbf files in the current directory are presented in a listbox and you must choose one or cancel the command. If the file has already been defined, an error message box is displayed and you may pick another file.

After a file has been selected, a series of dialog boxes guides you through the DataWorks file definition process.

In order to use the generic form edit available through a data display window, it is imperative that each field in the imported database be described.

File Print Command
A listing is produced of the main window file directory. Use the SETTINGS PRINT CONFIG function to check your printer configuration before printing the report.

File Directory Edit Menu
The Edit menu includes commands that enable you to initiate gross file actions (e.g., maintain index files) or edit the description of the file contained in the DataWorks file directory.

For more information, select the Edit menu command name.

Index
Pack
Description

Edit Index Command
Allow the creation of a new index file, the modification of an existing file, reindexing an existing index file, or the deletion of an index file. Indexing expressions may contain standard xBase functions. Dates should be indexed using the DTOS() function. Numbers should be indexed with the STR() function. Character fields should be made case insensitive by using the UPPER() function.

Edit Pack Command
When an xBase record is deleted, it is only *logically* deleted by having a flag placed in a special field that says "this record is deleted". The process of physically removing the deleted records from the file is known as *packing*.

All logically deleted records are physically removed from the file when the Edit Pack command is used. All associated index files are recreated after the pack operation is complete.

Edit Description Command

The description of the file as displayed in the main window file directory may be modified using this command.

File Directory Tools Menu
The tools menu includes commands that allow the user to design his own tabular displays.

For more information, select the Tools menu command name.

Tables

Tools Tables Command
You may define your own table display with the Tools Tables command. The contents of each column are completely under the user's control. You may use expressions to perform arithmetic calculations on one or more fields, concatenate strings, or even conditionally display data using the IIF() function.

Up to three dialog boxes may be presented when defining a user display table. Select one of the topics below for more information:

Table Select
Table Definition
Table Field Layout
Table Field Layout Pushbuttons

Table Select

If any user-defined tables for the highlighted file already exist, a dialog box is presented which allows you to select an existing table for editing, add a new table, or cancel the display altogether.

Table Definition
The table definition dialog box solicits the following information:

Table Name a 6 character user name that must be unique for this file. Use a descriptive name. For example, a simple table that displayed customer codes, names, and phone numbers could be called PHONES.

Description this is the window caption that will appear when this table is selected for display.

Index if any indexes for this file have been defined, their names will appear in a combo box. If the table is new, the default shown is the first index name (in alphabetical order). If the table has already been defined, the default is the last index chosen.

Col Head Color the color of the column heading may be selected from a combo box.

BOF expression A beginning of file expression (optional). For example, if you wished to begin the display with names beginning with the letter "G", the BOF expression would be LEFT(CUST_NAME,1)<"G".

EOF expression An end of file expression (optional). For example, if you wished to stop the display with names beginning with the letter "S", the EOF expression would be LEFT(CUST_NAME,1)>"R". Note that the BOF and EOF expressions are dependent on the sequence of the display (i.e., the index selected). The examples shown for BOF and EOF would only make sense if the file was indexed on CUST_NAME and that index was selected.

Filter expression A filter restricts the display to only those records that satisfy the filter condition. For example, to restrict the display to all customers whose names begin with the letter "F" through "Q", the file could be filtered with the following expression:

```
LEFT(CUST_NAME,1) >= "F" .AND. LEFT(CUST_NAME,1) <= "Q"
```

The filter expression is optional. Any filter defined here may be changed with the Filter command found in the data display window menu.

Scope Index This is a specialized field that determines the action to be taken when the HOME or END keys are depressed. The default value is 0, which takes you to the first and last records in the file respectively. If other than 0, then the HOME key will result in a seek on the file to the current key for the length specified by the scope index. The END key will seek to the portion of the key defined by the scope index (plus 1) and then skip back one record to position the record pointer to the last record in the group.

Always use 0 (zero) when the scope you are interested in is every record in the file or when every record in the file has a unique key. If the file is indexed with the possibility of a common prefix being part of many keys, and you wish to limit the display to this subset of records without requiring a filter expression, then enter the length of the common part of the key.

For example, if an accounts receivable subledger was indexed on CUST_CODE + INVOICE, then the common part of the key would be the CUST_CODE. There could be many records with the same CUST_CODE; the unique part of the key

would be the invoice number. We could set up the scope index as the length of the field CUST_CODE and define a BOF expression that started the display at a specific customer. The display would be limited to that customer and quick key positioning would be on the INVOICE portion of the key only if the Quick Index (discussed below) is set accordingly.

Quick Index sets the quick key status for indexed files. A 0 (zero) will turn quick key off. If an index is active, a number greater than zero will allow quick key searches. If the key to be used for the quick search starts at the first character position of the current index expression, use 1. If the quick key is to be built out of a preceding hidden part of the key plus the entered quick value (as in the CUST_CODE + INVOICE example shown above), use the length of the hidden key prefix plus 1 as the quick index. In the example, if the length of CUST_CODE is 6, use 7 as the quick index.

Table Field Layout
The final step in defining a custom display table is the specification of the fields (i.e., columns) you wish to view. A table field may be a specific field or an expression that combines more than one field or only shows part of a field or one that uses a conditional IIF() function to display different values dependent upon the truth of the conditional expression.

The list box at the top of the Table Field Layout dialog box contains all of the fields in the currently selected file. The data entry portion of the list box follows. It always starts out in Add mode (i.e., it assumes you wish to add another field or expression to the display table). If columns have already been defined, they appear in the list box at the bottom of the dialog. To edit an exisiting column definition, highlight the desired item and push the EDIT button, or double click the item to move it to the edit portion of the dialog. The following data entry is required:

Sequence — A left to right sequence number. The columns will be displayed from left to right beginning with the lowest sequence number. Each sequence number must be unique.

Column Head — Text displayed in the colum header box. This allows you to use more descriptive headings instead of the usually cryptic field names.

Type — Push the radio button that describes the type of data to be defined in the next box (either a Field or an Expression). If *Field* is selected, DataWorks verifies the existence of the field in the file. If *Expression* is selected, then the expression validity is tested. A field may be defined as an expression (which is to be avoided because it takes time to evaluate expressions) but an expression may not be defined as a field.

Field or Expr — Field names may be inserted into this edit control by at an item in the Fields list box and double clicking or by pressing the Select button when the desired field name is highlighted. The field name is always attached to the end of the text currently present in the control.

Expressions MUST evaluate as character strings. Use the STR() function to express numeric values as character strings. Use the DTOC() function or the DTOS() function to display date fields.

Table Field Layout Pushbuttons

Save	This button must be pushed to save the current field definition and move it to the table layout listbox.
Cancel	Cancels the current definition and reverts to Add mode.
Edit	Edits the highlighted item in the table layout listbox.
Delete	Deletes the highlighted item in the table layout listbox.
Exit	Terminates the dialog and returns to the main file select window.

File Directory Settings Menu
The Settings menu includes commands to toggle table displays between upper and lower case, select and define printer settings, and also set up security to dataworks files.

For more information, select the Settings menu command name.

Lowercase
Print Config

Settings Lowercase Command

Toggles the table display between upper and lowercase. If lowercase is not checked, the table display data is in the case as entered (not necessarily all uppercase). All lowercase results in a neater and more readable display.

Settings Print Config Command
Allows the selection of a windows defined printer and the ability to specify printer settings for the user selection. To change the active printer, you must use the Windows function Control Panel Printers found in the Main program group.

File Structure Edit Command

This menu item is only accessible from a displayed file structure window. It allows the user to alter the format of the database file.

File Structure Print Command

This menu item is only accessible from a displayed file structure window. It initiates a hard copy listing of the structure of the selected.file.

Data Table Edit Menu
The Edit command menu items allow updating, adding, or deleting records -- as well as editing a memo field if one is defined in the record structure of the selected file.

For more information, select the Edit menu command name.

Update
Add
Delete
Memo

Edit Update Command
The contents of the currently selected record are placed within a dialog box that is generated using the current structure definition of the file. If a record has never been updated in this file (or added to this file) then a new form is automatically generated and added to the DataWorks dictionary file DWFORMS.DBF.

Raw data edit utilizing this function verifies field contents as to type only (e.g., numbers are the only characters allowed within numeric fields, etc.).

Edit Add Command
An empty forms dialog box based upon the current structure definition of the file is displayed.

Raw data edit utilizing this function verifies field contents as to type only (e.g., numbers are the only characters allowed within numeric fields, etc.).

Edit Delete Command
The record sporting the current highlight is deleted. The user must confirm the delete before it actually takes place.

Edit Memo Menu
This command item is grayed (i.e., disabled) unless the existence of a memo field is detected when the browse table is set up.

The File menu allows saving text as a memo attached to the record, saving memo text as an independent ASCII text file, or importing text from a standard ASCII text file.

The Edit menu allows standard cut, copy, and paste functions on the memo text.

For more information, select the Memo menu command name.

File Menu
Edit Menu

Edit Memo File Menu
The File menu allows saving text as a memo attached to the record, saving memo text as an independent ASCII text file, or importing text from a standard ASCII text file.

For more information, select the File menu command name.

Save Memo
Import ASCII
Export ASCII
Print

Memo File Save Memo Command
Save memo text into .dbt file associated with this .dbf.

Memo File Import ASCII Command
Import an ASCII text file into the memo. The current memo contents will be overwritten.

Data may also be imported from other sources via the Windows Clipboard. See the Memo Editmenu commands for more information.

Memo File Export ASCII Command
Write the contents of the memo out as a standard ASCII text file.

Memos may also be exported to other targets (e.g., Windows Notepad workspace). See the Memo Edit menu commands for more information.

Memo File Print Command
Print the contents of the memo window.

Edit Memo Edit Menu
The Edit menu allows standard cut, copy, and paste functions on the memo text.

For more information, select the Edit menu command name.

Undo
Cut
Copy
Paste
Delete
Select All
Insert Date

Memo Edit Undo Command
All edits made to the current memo are undone (i.e., up to the last save).

Memo Edit Cut Command
Selected text is cut to the clipboard. The text is available for pasting into another area of the memo or into a document of another format (e.g., Windows Notepad).

Memo Edit Copy Command
Selected text is copied to the clipboard. The text is available for pasting into another area of the memo or into a document of another format (e.g., Windows Notepad).

Memo Edit Paste Command
If any text currently resides in the clipboard, this menu item is enabled. Position the text cursor to the point of insertion and select this command to paste the clipboard text into the memo.

Memo Edit Delete Command
Selected text is deleted. If you wish to retain the text for pasting elsewhere, use the Memo Edit Cut command instead.

Memo Edit Select All Command

Use this command to select all of the text in the memo window. The selected text may be acted upon with the Delete, Cut, and Copy commands.

Memo Edit Insert Date Command

This command creates a current date-time stamp string and inserts it into the memo text at the edit cursor position. The string is also placed into the clipboard and is available for pasting into other memos or documents.

Data Table Query Menu
Dataworks can locate any string input by the user if it exists in the database. The query is neither case sensitive nor field specific.

For more information, select the Query menu command name.

Find Next
Search

Query Find Next Command
This item is grayed (i.e., disabled) until a string is entered with the Query Search Command. Once a string has been entered, the next occurrence of that string in the database is found. The record highlight is moved to the line containing the search string.

Query Search Command
A dialog box is presented allowing the user to specify a search string (characters or numbers). A DataWorks Search is neither case sensitive nor field specific. If the search string is found, the record highlight is moved to the row containing the string.

Data Table Print Command
Prints the currently selected table. If the width of the record is greater than the defined max width, the listing is truncated on the right.

Data Table Filter Command
The user may define a data Filter, which restricts the display to a subset of the data depending on the conditions imposed by the user. An example of a filter would be to restrict the display to all customers whose names begin with the letter 'A'. Filter expressions must always evaluate to a logical .TRUE. or .FALSE. through the use of relational operators.

A filter expression is built according to standard xBase syntactic rules. In the example above, if the field holding the customer's name was CUST_NAME, the filter expression would be LEFT(CUST_NAME,1) = "A".

Data Table Join Command
The user may Join one or more databases together based upon some common value found in every file in the defined relationship.

Any open file can be joined to any other provided that a field in the first file can be used as a key into the target file. The target file must be indexed and have as its key (or at the beginning of its key) a value that could match the data in the field defined as a join link in the first file.

For example, a customer master file could be opened. A field in the master file is CUST_CODE. This field is used to set up a relationship to an accounts receivable subledger which has an index built on AR_CODE + AR_INVOICE. The field AR_CODE in the subledger is derived from the CUST_CODE field in the master file. Once the subledger file is opened, another relationship could be set up to an invoice file using the AR_INVOICE field from the subledger to an invoice file that also contains this value and that has an index built using its invoice number field as the key.

Whenever the record pointer is moved in any of the joined files, a corresponding move is made in the files it is joined to that are below it in the chain - automatically! Data in joined windows is filtered to the defined key values. To use the example above, if customer 123456 is highlighted in a top window, only subledger records for 123456 will be displayed in the first joined window, and only details for the invoice highlighted in the subledger window will be displayed in the third window.

Only one join definition per window is allowed. Closing a window in a chain of joined windows closes all windows that were joined below the one being closed.

Join Definition
When a join request is made, a dialog box is opened that contains a list of all past joins defined for this file. One of these may be selected, or a new join may be defined by picking an indexed file from the second listbox and a field to be used as key from the third listbox.

Once a join has been established, the JOIN menu item is grayed to indicate that it is no longer possible to join another file to this one.

If the joined file has a user defined table associated with it, you may choose to display the join window using this format. If a table is selected, its name is attached to the join definition and in future will be used whenever this relationship is specified. Each dbf-ntx pair defined in a relationship must be unique. If you have already defined a join using a specified index with a table, and you wish to display the data in some other format, the first join definition must be deleted.

File Directory Window
All files that have been defined to the DataWorks dictionary have their names and descriptions displayed in this window. This window is the main window and is always present on the screen.

The data in a defined file may be displayed by positioning the inverse video bar over the file name and pressing the *ENTER* key, double clicking the left mouse button on a file name, or by selecting the FILE OPEN command from the FILE menu.

Multiple files, structures, etc. may be displayed in multiple windows by reactivating this screen by clicking any portion and then selecting another file (or even the same one again).

Data Window

Multiple data tables and file structure displays may be present on the Windows desktop. New files may be opened or their structures displayed by pointing at the main window and selecting another file from the directory display.

Memo Window
This is a standard text editing window. Changing its size will automatically reformat all text within the window. Dynamic Data Exchange through the Windows clipboard is active.

Maximize Icon
Expands an application window to the full size of the screen.

To choose this command, do one of the following:
(1) click the maximize icon;
(2) click the System Menu icon in the upper left corner of the application window and then click Maximize.
(3) Press Alt, Spacebar, X.

Minimize Icon
Shrinks an application window to an icon.

To choose this command, do one of the following:
(1) click the minimize icon;
(2) click the System Menu icon in the upper left corner of the application window and then click Minimize.
(3) Press Alt, Spacebar, N.

Sizing Border

The size of the window may be altered by pointing at the sizing border with the mouse and holding the left button down. The mouse cursor will change. Drag the outline border that appears to a new position and release the button.

System Menu
Every DataWorks window has a system menu accessed by pointing at the bar in the upper left corner of the window. Every DataWorks display window is closed using the system menu. If the main file directory window is closed, DataWorks terminates.

The system menu may also be accessed through the keyboard with ALT-SPACEBAR. A window may be closed with ALT_F4.

Title Bar
A window may be moved by placing the mouse cursor on the window title bar and pressing and holding down the left mouse button. With the button down, a window outline appears that may be dragged to a new location on the windows desktop and dropped.

Defined Terms
character fields
clipboard
date fields
dbf
field
file
functions
index
keys
logical fields
memo fields
numeric fields
ntx
relational database
xbase

Error Messages

100 **Error Creating File**. An illegal file name has been specified, the file already exists, or the disk is full.

120 **Error Opening File**. The file does not exist, or the number of files specified in the CONFIG.SYS FILES= statement has been exceeded.

140 **Error Reading File**. Probable disk failure.

160 **Error Writing File**. Disk is full or disk has failed.

180 **Error Closing File**. Possible file allocation table corruption.

190 **Error Removing File**. Incorrect file name, file does not exist, or possible disk failure.

200 **File is not a DataBase**. The specified file does not conform to xBase specifications. If the file is a known xBase database, the xBase header has probably been corrupted.

260 **Record Length is Too Large**. The maximum length of any given record is 32666.

270 **Unrecognized Field Name**. The requested field does not exist in the current database. There has either been a spelling error, the database header is corrupted, or a request was made to display a database using a defined table and a field that was present when the table was originally built has since been deleted.

300 **Error Building Index File**. There is insuffucient memory available to build the requested index.

310 **Error Closing Index File**. Possible file allocation table corruption.

320 **File is not an Index File.** This version of DataWorks supports Clipper NTX files only. Either the file does not conform to these specifications or its header has been corrupted.

330 **Index File is out of date**. The index file key for a specific database record does not exist. Use the Edit Index command to rebuild the file's indices.

335 **Index File Record does not Exist.** The database record for the corresponding index key does not exist. Use the Edit Index command to rebuild the index.

350 **Key Evaluates to a Logical Result**. All keys must evaluate to character strings.

360 **Key Length or Type has Changed**. A field referenced in a key expression has had its length or type changed. The index is no longer valid. Use the Edit Reindex command to recreate the index.

370 **Key Length over 100 Characters**. The index key expression evaluates to a length greater than 100 characters, which is the maximum length allowed.

380 **Seek on Database with no Index File**. The index select area does not match the current database. Check your index file definitions with function Edit Index.

400 **Locking a File**. A file conflict has arisen when a file lock was requested with DataWorks. Retry the operation.

450 **Unlocking a File**. A problem occurred while unlocking a database or index file. Retry the operation or exit DataWorks and try the same operation again. A network confict may be at fault.

500 **Database not Located while Evaluating Expression**. The database specified in the expression was not located.

510 **Executing Null Expression**. The current expression could not be evaluated because of a memory addressing problem.

515 **Illegal Date**. Illegal date value was encountered when seeking or evaluating an

expression that contains a date. use the DTOS() or DTOC() functions to ensure consistency and reliability. Standard date entry format is MM/DD/YY.

520 **Expecting "," or ")" while Evaluating Expression.** A comma parameter delimiter or a right parenthesis is missing from the expression.

530 **Expression is not Complete.** Error evaluating incomplete expression.

540 **Overflow while Evaluating Expression**. The result of the expression was either too large or the expression was too complex to evaluate.

550 **Parameter or Operator has the wrong Type**. Function parameters have data type requirements (.e.g., SUBSTR(123456,2,4) is illegal because the first parameter must be of type character). Operators also require the same data type on each side of the operation (e.g., 3 + "ABC" is illegal; both types must be either numeric or character).

560 **Right Parenthesis Missing in Expression**. Un unpaired set of parenthese has been detected in an expression.

570 **Unrecognized Function in Expression.** The function entered in the expression is not supported by DataWorks. Check the functions listed in this help file.

575 **Unrecognized Operator in Expression**. An illegal operator has been detected in an expression. See the list of valid operators listed in this help.

580 **Unrecognized Value in Expression.** An item in an expression cannot be evaluated as a string, field , number, logical value, or function.

590 **Unterminated String in Expression**. A left delimiter in the expression does not have a corresponding reight delimiter. String delimters are either single or double quotation marks.

595 **Wrong Number of Parameters in Expression**. An expression function or operator was given an illegal number of parameters.

900 **Out of Memory**. No more Windows memory available.

1000 **Not Enough Memory to Sort.** An attempt has been made to create an index for a very large file. The attempt has failed because of a lack of memory. Try reducing the number of key elements in the index expression.

1100 **Physical file exists. Use FILE IMPORT**. An attempt was made to create a file with the FILE NEW command. The name of the new file already exists in the defined data directory. Either change the name if the intent is to create a new file, or use the FILE IMPORT command to define the existing file to DataWorks.

1101 **Invalid character in file name**. xBase file names must begin with a letter and may not contain embedded blanks or a hyphen.

1102 **Invalid character in field name.** An xBase field name must begin with a letter. The remainder of the field name may contain letters or numbers or the underscore (_) character. The only special character allowed in a field name is the underscore.

1103 **File already defined.** An attempt has been made to import a file structure for a file that has already been defined to DataWorks.

1104 **File must be described**. A new file or imported file must have a title entered.

1105 **File name must be defined.** The DOS file name must be entered for a new database file.

1106 **Numbers only allowed.** The dialog box control can only accept numeric values (usually including decimal points and signs).

1107 **Field name required**. A new field definition requires a field name.

1108 **First letter must be alphabetic**. Field and file names must begin with an alphabetic character.

1109 **Duplicate field name.** The field name entered already exists in the file structure.

1110 **Password not on file. Aborting.** The standard password is MASTER. It may be changed by altering the dw.ini file.

1111 **Fields for selected file not found**. A file has been defined to the DataWorks file directory, but its field structure is missing from the dictionary. Delete the file name entry and import the file again.

1112 **Search string not found**. Self explnatory.

1113 **Sequence already on file.** The field sequence number assigned to the current field has already been assigned to a different field.

1114 **Field must have length**. A zero length field is not allowed.

1115 **Max numeric length is 19**. The maximum width of a numeric field is 19 characters including the decimal point and sign position (if negative numbers allowed).

1116 **Max decimals is field length - 2**. The numeric field may not have more decimal positions than the length of the field minus two (one for the decimal and one for a leading zero).

1117 **File creation Error**. The file could not be created, probably because of a disk problem or a network security violation.

1118 **File open in another window. Close it first**. An attempt has been made to perform an operation on a file that requires its exclusive use, and the file is open in another DataWorks window.

1119 **File dwindex.dbf missing. Aborting**. One of the required DataWorks dictionary files is missing from this directory.

1120 **File dwindex.ntx missing. Aborting.** See 1119 above.

1121 **Index expression required.** An xBase expression is required to build an index.

1122 **File name required**. The new index must have a DOS file name assigned.

1123 **Index name already defined**. The new index name must be unique in the dictionary file.

1124 **Index make error**. The index creation did not take place. See 1117 above.

1125 **Index file already exists.** The DOS file name of the new index file already exists in this directory.

1126 **No memory for dialog box.** We have run out of Windows memory.

1127 **Max field length is 1024**. The defined field length is too long.

1128 **Date format error. MM/DD/YY required.** A data field entry does not conform to the DataWorks required format.

1129 **Logical values ynYNtfTF only allowed**. Input to a logical field must conform to the defined values.

1130 **Too many signs**. A numeric field may only have one sign, and that must be in the first position.

1131 **Sign must be in first position**. See 1130 above.

1132 **Too many decimals**. A numeric field may have only one decimal point.

1133 **Form edit active. Finish it first**. Complete the record edit operation before attempting this operation again.

1134 **Cannot allocate memory for memo edit.** Max memo size is 32k and we don't have enough Windows memory to edit a memo.

1135 **Edit control out of space**. We are out of WIndows memory. An edit control has requested more memory and has been refused.

1136 **File dwfields.dbf missing. Aborting**. See 1119 above.

1137 **File dwfields.ntx missing. Aborting**. See 1119 above.

1138 **File does not exist.** A file selected from the DataWorks directory does not exist in the current directory.

1139 **Selected index does not match dbf.** The key expression in the selected index does not contain references to fields in the matching dbf. This index is not for this dbf.

1140 **Close field(s) windows first**. Close the file structure displays before attempting this operation again.

1141 **Delete not allowed on DW system file**. An attempt was made to delete a DataWorks dictionary file.

1142 **Help not available for menu item.** I missed something.

1143 **Expression must evaluate as logical TRUE or FALSE**. An entered filter expression must return a TRUE or FALSE value, not any other data type.

1144 **No records found that pass filter.** As it says. No records satisfied the filter condition. The display restarts at the top.

1145 **Unique table name required**. The table name entered has already been defined.

1146 **Table description required**. The table must have a title.

1147 **File dwtabfld (dbf or ntx) does not exist**. See 1119 above.

1148 **Unable to evaluate expression.** Something is wrong with the entered xBase expression. A previous message should have told you exactly what. This message marks the passing of the evaluation attempt.

1149 **Expression must evaluate as character string**. DataWorks index keys must evaluate as character strings. Use the STR() function to convert numbers to strings and the DTOS() function to convert dates to strings.

1150 **Out of memory.** Self explanatory.

1151 **Table definition file open. Close first.** DO as it says before retrying the current operation.

1152 **Relations file DWRELATE (dbf or ntx) not found**. See 1119 above.

1153 **Join dialog active. Complete or cancel first**. Dispose of the dialog box before retrying the selected operation.

1154 **File-Index pair already defined.** The new relationship being defined is already on file.

1155 **No records found that match join key.** A join display failed because there were no matching records. Information only. Try selecting a different controlling record.

1162 **Cannot edit result of expression.** The user double clicked a table field that is being displayed as the result of an expression in a table definition. We obviously cannot edit this.

1163 **Use Edit Memo menu item for memos**. Use the appropriate function instead of double clicking the memo field entry.

1164 **Field edit allowed on data windows only**. An attempt was made to double click an item on a file structure display.

1165 **Field edit not allowed on joined windows**. Only fields that belong to the parent window can be edited onscreen.

1166 **Only one active field edit allowed**. Dispose of the active one before trying again.

1167 **Printer error!** Self explanatory.

1168 **Invalid registration number**. If you have registered DataWorks, try again. Maybe you entered it wrong. Otherwise, call for a new number (or the correct one).

1169 **User aborted print job.** Information only. Self explanatory.

Windows Keys

Choose from the following list to review the keys used in Windows:

Cursor Movement Keys
Dialog Box Keys
Editing Keys
Help Keys
Menu Keys
System Keys
Text Selection Keys
Window Keys

Cursor Movement Keys

Key(s)	Function
DIRECTION key	Moves the cursor left, right, up, or down in a field.
End or Ctrl+Right Arrow	Moves to the end of a field.
Home or CTRL+Left Arrow	Moves to the beginning of a field.
PAGE UP or PAGE DOWN	Moves up or down in a field, one screen at a time.

Dialog Box Keys

Key(s)	Function
TAB	Moves from field to field (left to right and top to bottom).
SHIFT+TAB	Moves from field to field in reverse order.
ALT+letter	Moves to the option or group whose underlined letter matches the one you type.
DIRECTION key	Moves from option to option within a group of options.
ENTER	Executes a command button. Or, chooses the selected item in a list box and executes the command.
ESC	Closes a dialog box without completing the command. (Same as Cancel)
ALT+DOWN ARROW	Opens a drop-down list box.
ALT+UP or DOWN ARROW	Selects item in a drop-down list box.
SPACEBAR	Cancels a selection in a list box. Selects or clears a check box.
CTRL+SLASH	Selects all the items in a list box.
CTRL+BACKSLASH	Cancels all selections except the current selection.
SHIFT+ DIRECTION key	Extends selection in a text box.
SHIFT+ HOME	Extends selection to first character in a text box.
SHIFT+ END	Extends selection to last character in a text box

Editing Keys

Key(s)	Function
Backspace	Deletes the character to the left of the cursor. Or, deletes selected text.
Delete	Deletes the character to the right of the cursor. Or, deletes selected text.

Help Keys

Key(s)	Function
F1	Gets Help and displays the Help Index for the application. If the Help window is already open, pressing F1 displays the "Using Windows Help" topics. In some Windows applications, pressing F1 displays a Help topic on the selected command, dialog box option, or system message.
SHIFT+F1	Press this key combination and then choose a command, click the screen region, or press a key or key combination you want to know more about. (This feature is not available in all Windows applications.)

Menu Keys

Key(s)	Function
Alt	Selects the first menu on the menu bar.
Letter key	Chooses the menu, or menu item, whose underlined letter matches the one you type.
Alt+letter key	Pulls down the menu whose underlined letter matches the one you type.
LEFT or RIGHT ARROW	Moves among menus.
UP or DOWN ARROW	Moves among menu items.
Enter	Chooses the selected menu item.

System Keys

The following keys can be used from any window, regardless of the application you are using.

Key(s)	Function
Ctrl+Esc	Switches to the Task List.
Alt+Esc	Switches to the next application window or minimized icon, including full-screen programs.
Alt+TAB	Switches to the next application window, restoring applications that are running as icons.
Alt+PrtSc	Copies the entire screen to Clipboard.
Ctrl+F4	Closes the active window.
F1	Gets Help and displays the Help Index for the application. (See Help Keys)

Text Selection Keys

Key(s)	Function
SHIFT+LEFT or RIGHT ARROW	Selects text one character at a time to the left or right.
SHIFT+DOWN or UP	Selects one line of text up or down.
SHIFT+END	Selects text to the end of the line.
SHIFT+HOME	Selects text to the beginning of the line.
SHIFT+PAGE DOWN	Selects text down one window. Or, cancels the selection if the next window is already selected.
SHIFT+PAGE UP	Selects text up one window. Or, cancels the selection if the previous window is already selected.
CTRL+SHIFT+LEFT or RIGHT ARROW	Selects text to the next or previous word.
CTRL+SHIFT+UP or DOWN ARROW	Selects text to the beginning (UP ARROW) or end (DOWN ARROW) of the paragraph.
CTRL+SHIFT+END	Selects text to the end of the document.
CTRL+SHIFT+HOME	Selects text to the beginning of the document.

Window Keys

Key(s)	Function
ALT+SPACEBAR	Opens the Control menu for an application window.
ALT+Hyphen	Opens the Control menu for a document window.
Alt+F4	Closes a window.
Alt+Esc	Switches to the next application window or minimized icon, including full-screen programs.
Alt+TAB	Switches to the next application window, restoring applications that are running as icons.
Alt+ENTER	Switches a non-Windows application between running in a window and running full screen.
DIRECTION key	Moves a window when you have chosen Move from the Control menu. Or, changes the size of a window when you have chosen Size from the Control menu.

Character Fields
A character field may contain any sequence of characters or numbers to a maximum length of 64k (65,535 characters).

Clipboard
The Windows "clipboard".is the prime mechanism of Dynamic Data Exchange in the Windows environment. Many Windows programs may either copy or cut items into the clipboard. Items placed in the clipboard may then be pasted into other documents or records in the same application or even different applications. For example, text written with the Windows Notepad may be copied into the clipboard and then pasted into a DataWorks memo.

Date Fields
Date fields are entered, displayed, and reported as MM/DD/YY. The data representation within the record is YYYYMMDD. Date fields used within index expressions should be defined with the function DTOS(date_field) which converts the date into YYYYMMDD format and properly organizes the index with the oldest dates first.

.dbf
This term refers to the standard DOS physical extension given to xBase data files (dbf = database file).

Field

Database data is normally displayed in the form of tables. Each row in the table represents a database record, and each column represents a field. Each field within a record must have a unique name (up to 10 characters long) and begin with a letter. The field name may contain letters or numbers. The only special character allowed within a field name is the underscore (_). A field must also have its type (character, numeric, date, logical, memo) and length defined. Fields with a set length (dates and memo references) have their lengths defined by DataWorks.

File
A file is a collection of related data that is accessed by supplying a physical file name that matches a name in a disk directory which points to the location of the data on the disk. DOS file conventions allow eight characters in a file name with an optional three character extension. DataWorks file names must begin with a letter and may not contain embedded blanks or a hyphen.

Functions
xBase functions modify data. They are used in index expressions, filter expressions, relational expressions, and data replacement expressions. For example, the xBase command REPLACE ALL CUSTCODE WITH UPPER(CUSTCODE) would transform all data in all fields named CUSTCODE in the currently selected file into uppercase.

Index
Raw data is record in a .dbf file in the order in which it is entered. To properly organize the date for retrieval it is necessary to build an index file based upon defined values within the data record.

A data file may be indexed on one or more fields. The fields may be modified by xBase functions. For example, if a sales transaction file was to referenced in customer-date order, and the names of these fields were CUSTCODE and SALEDATE respectively, then an index file could be built using the expression UPPER(CUSTCODE)+DTOS(SALEDATE). The data in the customer code field is modified with the UPPER() function which converts all characters in the field to uppercase. This ensures the correct collating sequence and simplifies retrieval. The date field is modified with the DTOS() function, which converts a standard MM/DD/YY string into the format YYYYMMDD, which ensures that the oldest dates appear first. The plus sign in the expression concatenates (i.e., joins) the two fields together to produce a sequenced index into the data.

An index expression may not result in an index key longer than 250 characters. A dbf file may have more than one index associated with it. Clipper conventions assign the extension .ntx to index files. These index files are not compatible with dBase III files.

Index Keys
An index key is the result of an index expression. A key may be composed of the data in one or more fields and may even be combined with literal values. The data may also be modified with xBase functions.

For example, if a sales transaction file was to referenced in customer-date order, and the names of these fields were CUSTCODE and SALEDATE respectively, then an index file could be built using the expression UPPER(CUSTCODE)+DTOS(SALEDATE). The data in the customer code field is modified with the UPPER() function which converts all characters in the field to uppercase. This ensures the correct collating sequence and simplifies retrieval. The date field is modified with the DTOS() function, which converts a standard MM/DD/YY string into the format YYYYMMDD, which ensures that the oldest dates appear first. The plus sign in the expression concatenates (i.e., joins) the two fields together to produce a sequenced index into the data.

An index expression may not result in an index key longer than 250 characters.

Logical Fields
Logical fields are one character fields that indicate TRUE or FALSE and may contain the characters T, F, Y, N and their lowercase equivalents.

Memo Fields

Memo fields are textual notes attached to a data record. They are variable in length. DataWorks supports memos to a maximum of 32,767 (32k) characters. Memos are kept in files that have the same name as the related .dbf but with a file extension of .dbt.

Numeric Fields

A numeric field may contain up to 19 characters (including an optional decimal point and optional minus sign). The numeric character set is composed of the numbers 0 through 9, - (minus), and an optional decimal. Signs must precede the numeric portion of the field. The sign of the number is positive unless there is a preceding minus sign. The maximum number of digits to the right of the decimal point is 16. A field that contains a decimal point must have at least one digit defined to the left of the decimal point. For example, a field that may hold a maximum value of .999 must be defined with a length of 5. Its display representation is 0.999.

.ntx
This term refers to the Clipper standard physical extension given to xBase index files.

Relational Database
The term *relational* refers to the fact that data in one database may be related to data in another database. The ability to relate data from table to table greatly increases your organizational abilities and reduces data redundancy. Relationships among files are defined by linking like fields between the files.

For example, you could have a customer database that includes a customer identification field, name, address, etc. Another potential file would be an inventory file that includes an inventory code number to identify each item, an item description, quantity on hand, price, etc. The customer file and inventory file would be termed master files and would have associated index files built upon the customer and inventory codes respectively. Each record in a master file must have a unique key.

A third file could be inventory items sold to customers. It would include the customer code, inventory code, quantity sold, date sold, and perhaps price. This is a transaction file. It would probably contain two index files - one on customer+inventory code, and the other on inventory+customer code. Notice that this file does not require unique keys. It also does not require all the extraneous information contained in the customer file or the inventory file. To extract the customer name associated with a sale it is only necessary to define a relationship between the customer code kept in the transaction file and the customer code in the master file.

xBase
This is a generic term describing any relational database management system modeled after the structures laid down by dBase III or dBase III plus.