ISAM FILE MANAGER REFERENCE

JUNE 1984

2535 Garcia Avenue
Mountain View, CA 94043
(415) 961-4800

Manual Name: ISAM File Manager Reference Manual
Order Number: 29200-44
Issue date: June 1984

TABLE OF CONTENTS

FIGURES

TABLES

ABOUT THIS BOOK

The ISAM file Manager is a set of procedures and functions that you use to create and manipulate indexed, sequential-access data files. (ISAM refers to the file access method -- the Indexed, Sequential-Access Method.) The procedures are written (and described here) in the Pascal-86 language, but they are usable from programs in other GRiD supported languages.

The first chapter talks generally about the procedures and functions and what they do. It also describes the structure of the data files created. Chapter 2 explains the format this book uses to give you procedure information. Chapter 3 lists the detailed procedures alphabetically for easy look-up.

Appendix A contains the error codes (and associated messages) returned by the status parameter. Appendix B gives you run-time information (names of files to load and include). For general information on the programmer's interface, loading, linking, and compiling, see the GRiD Program Development Guide and the appropriate language manual.

The ISAM File Manager comprises both procedures and functions. (A function is like a procedure except that it returns information "in its name," in addition to information returned by its parameters.) Nevertheless, this manual frequently uses the word "procedures" to refer to a combination of procedures and functions.

CHAPTER 1: OVERVIEW OF THE ISAM FILE MANAGER

This chapter provides a quick overview of the capabilities and operation of the ISAM File Manager.

PROCEDURES FOR WORKING WITH INDEXED DATA FILES

The ISAM File Manager allows you to create and access indexed (keyed) data files programmatically, in essence, lets you build and use your own database. Figure 1-1 loosely depicts the relationship between your program, the ISAM File Manager, and the data and key files you create. Table 1-1 summarizes what the procedures do.

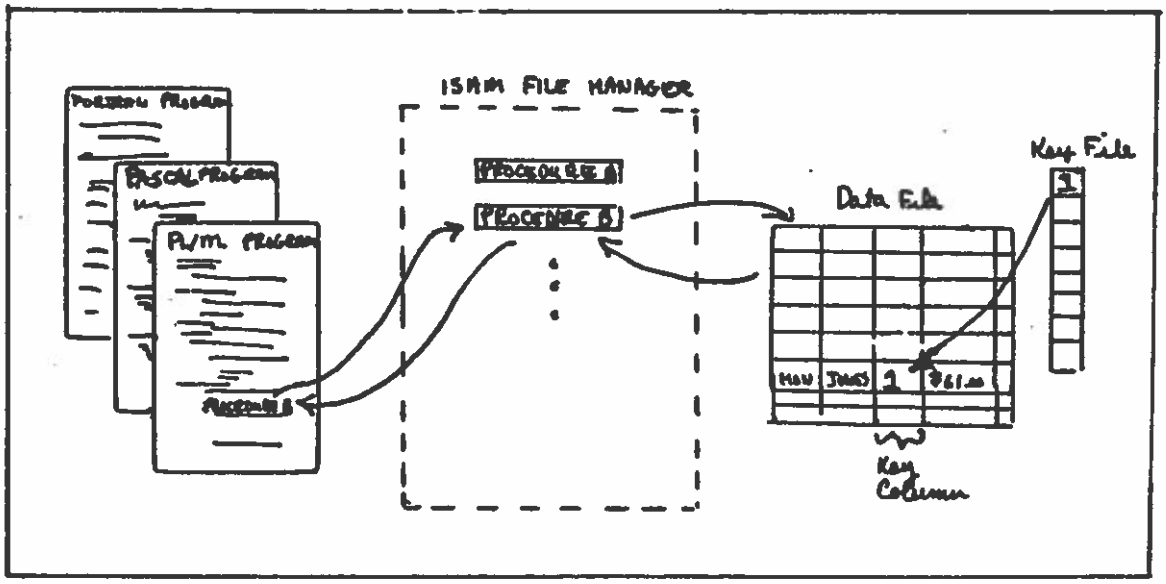


Figure 1-1. Programmatic Access of Data Files

Table 1-1. Summary of Procedures

ISAMCreate	Creates a data file in which all records have the same number of fields (columns) and the fields are of variable length.
ISAMAttach	Makes a data file and its associated key files available to your program.
ISAMOpen	Opens and allocates memory for an attached data file.
ISAMCreateIndex	For an open data file, designates a column to be a key column. The file manager creates a key file containing the alphabetically or numerically sorted values in that column.
ISAMOpenIndex	Opens and allocates memory for a key file.
ISAMAppendRecord	Adds a record to the end of a data file. When there are no records, adds the first one.
ISAMReplaceRecord	Replaces the content of a record in a data file.
ISAMDeleteRecord	Deletes the content of a record from a data file (marks it deleted).
ISAMChangeColumns	Inserts, deletes, duplicates or moves columns in an open data file.
ISAMRead	Copies a record from an open data file.
ISAMFind	Finds the first record with a certain value in a certain column and returns the I.D. of the record.
ISAMFindNext	Finds the next record and returns the content of a specified column and the record I.D.
ISAMFindGeneric	Finds the first record with a column value greater than or equal to a certain value and returns the value and the record I.D.

Table 1-1 (Continued). Summary of Procedures

ISAMResetFile	Moves the file pointer to the beginning of a data file or to the first key in a key file.
ISAMGetStatus.	Returns the number of columns, number of key columns, and number of records deleted or replaced since the last compression.
ISAMCompress	Physically removes records marked deleted from an open data file.
ISAMDecodeError	Returns a message explaining an error code.
ISAMCloseIndex	Closes a key file and deallocates the memory associated with it.
ISAMDeleteIndex	Deletes a key file associated with a data file.
ISAMClose	Closes and deallocates memory assigned to a data file.
ISAMDetach	Makes a data file and its associated key files unavailable to your program.
ISAMExit	Closes and detaches data and key files.

USING THE PROCEDURES

Your language manual contains specific information on calling procedures from your particular language. Since you need to pay attention to differences between your program language and Pascal-86, such as differences in data types, we recommend you see that information. For example, if you are programming in Fortran-86, see the Fortran-86 User's Guide, Appendix H.

In general, you use the ISAM File Manager procedures to create and load data files, then perform quick, keyed access.

Your first task is to create the data file (ISAMCreate). If you are going to use it immediately, you allocate memory space to it with a call to ISAMOpen. To use a file that is already created, you need only attach (ISAMAttach) and open it. (The attach associates the already created file with your program.)

Both ISAMCreate and ISAMAttach return a file number (fileNumber) that you use in all other procedure calls to identify the data file.

Your second task is to determine which, if any, columns of the

The ISAM File Manager takes the values in that key column, sorts them alphabetically or numerically depending on the column format, and places them in a key file. Each key in the key file is associated, via a record I.D., with the appropriate data file record. Later, when you search the column (ISAMFind or ISAMFindGeneric) in preparation for some activity, the sorted order of the key file makes the search go faster, especially if the data file contains a large number of records.

You may designate that the key column accept only unique keys. When keys are nonunique, the search finds the first occurrence and returns an indication of whether the next record is different; if it isn't, you must call ISAMFindNext to get the record I.D. of the second occurrence.

The third task is the real work. Use ISAMAppendRecord to add content to the data file; ISAMFind, ISAMFindNext, and ISAMFindGeneric to locate records in various ways; ISAMResetFile to reset the file pointer; and ISAMReplaceRecord, ISAMRead, ISAMDeleteRecord, ISAMChangeColumns to manipulate the records. Key files are automatically updated when you alter records in the data file.

The fourth and last task, as you may guess, involves shutting down work. Use ISAMGetStatus to return the number of records actually deleted, then decide whether to compress the file or not. ISAMCompress physically removes records you deleted (marked deleted), but it takes a little time. ISAMClose deallocates the memory space allocated to the data file, and ISAMCloseIndex does the same thing for a key file; you may wish to use these closes to save space while you keep the files "in the wings." ISAMDetach closes AND makes the data file (AND its key files) unavailable to your program; ISAMExit does everything ISAMDetach does AND does not return to your program.

The set of procedures you use on any occasion depends, of course, on what you want to do; the two sequences below illustrate typical tasks.

To create an indexed data file and enter data in it, you might call this sequence of procedures.

- ISAMCreate, to create your data file (attachment is automatic).
- ISAMOpen, to open the file and allocate memory space.
- ISAMCreateIndex, to establish a field (column) as a key column. The ISAM File Manager creates a sorted key file of the values in the key column.
- ISAMOpenIndex, to open the key file and allocate it memory space.
- ISAMAppendRecord, to insert the first record.
- ISAMAppendRecords, to insert subsequent records.
- ISAMDetach, to close and detach the data file and its associated key file.

Later, to amend the same database, you might use the following sequence of procedures:

- ISAMAttach, to make the data file and key file available to your program.
- ISAMOpen, to open and allocate memory space to the data file.
- ISAMOpenIndex, to open and allocate memory space to the key file.
- ISAMFindGeneric, to find a record in a key column.
- ISAMReplaceRecord, to replace the found record.
- ISAMChangeColumns, to delete a column.
- ISAMFind, to find a record with a certain value in a column.
- ISAMDeleteRecord, to delete the found record.
- ISAMExit, to detach and close the data and key files.

FILE STRUCTURE

The data file that you create (a call to ISAMCreate) is conceptually a matrix in which each row will hold a record and each column contains related record fields. You do not designate the length of the record fields; instead, the ISAM File Manager dynamically handles any length field. Figure 1-2 shows the empty matrix.

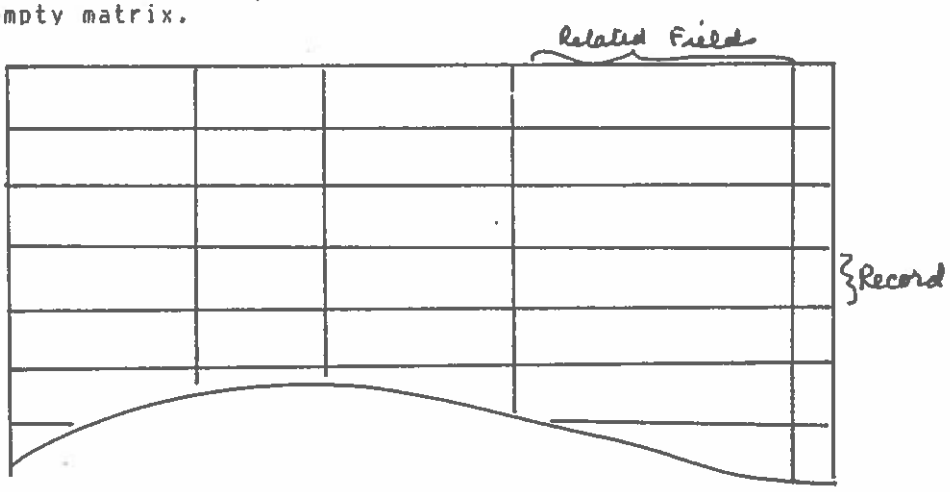
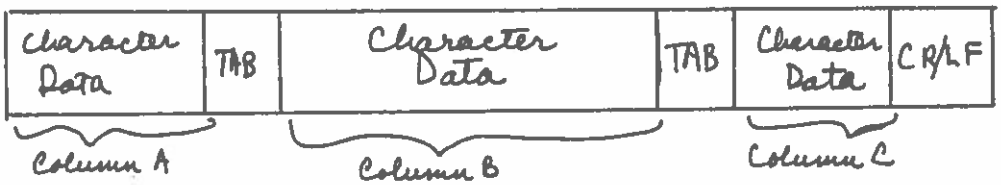


Figure 1-2. The ISAM Data File Structure

To add content to the database, you call ISAMAppendRecord for each record to be added. The input parameter NewRecord is a pointer to a string containing the record. The data in the string must be of the form



The ISAM File Manager takes each TAB character to represent the end of data in a column and each CARRIAGE-RETURN/LINE-FEED (CR/LF) to represent the end of the record. The record you append must have the correct number of columns for the data file.

Not coincidentally, this file structure is compatible with GRiD System's "Interchange File Format." The Interchange File Format makes it easy to move data from one application to another, for example from a GRiDPLAN worksheet to a GRiDFILE database. It is possible that you may wish to access records from an application, for example from a GRiDFILE database, using an ISAM application program of your own. When you do so, these records will contain TABs and CARRIAGE-RETURNS/LINE-FEEDs, leaving you to ensure only that the records contain the correct number of columns for the destination database.

application program of your own. When you do so, these records will contain TABs and CARRIAGE-RETURNS/LINE-FEEDs, leaving you to ensure only that the records contain the correct number of columns for the destination database.

CHAPTER 2: UNDERSTANDING THE PROCEDURE INFORMATION

This chapter explains what you need to know to call a procedure. You need to understand the procedure's operation and its return value, and you need to know the parameters, their order, their pass methods and directions, and their data types. Figure 2-1 shows the ISAMCreate function in the format this manual uses to give you procedure information.

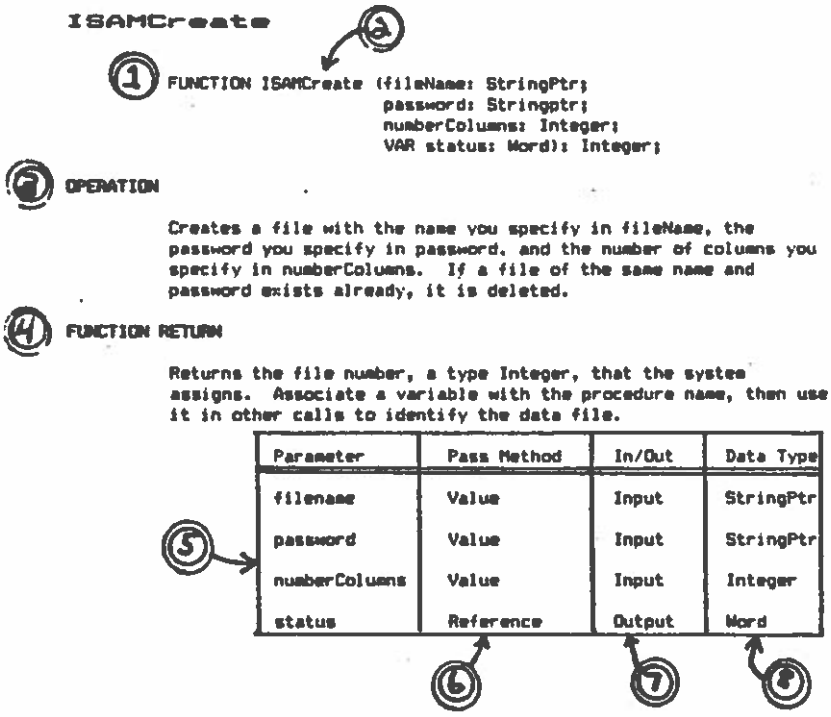


Figure 2-1. Procedure (Function) Format

Pascal Procedure (Function) Declaration

This is the declaration in the Pascal-86 language. If you are a Pascal programmer, you construct your call to the procedure by referring to this declaration. It tells you the procedure name and the order and types of the parameters.

Procedure (Function) Name

Procedures are listed by name in Chapter 3. You use the name in your call to the procedure.

Procedure (Function) Operation

The procedure operation describes what work the procedure does.

Function Return

Functions pass information back to your program in the form of a function return. It is up to you to associate a variable with the function name. The variable accepts the output information. In ISAMCreate, a system assigned file number of the data type Integer is returned. Do not confuse the function return with ouput parameters (discussed below) which also return information to your program.

Parameter Names

In the table of parameter information, the first column contains procedure parameter names in their correct calling order. Each time you call the procedure, parameters in your call (actual parameters) are matched to the procedure parameters -- your first parameter to the procedure's first, the second to its second, and so on. Your call to the procedure, then, must contain actual parameters in the same order as the procedure parameters.

The Status parameter is the last one for almost every procedure. It returns information on the success of the operation (see Appendix A for a list of the status return numbers and their meanings).

Pass Method

The second column tells you a parameter's pass method, how it passes information to the procedure, or, in the case of an output parameter, how the procedure passes information to your program.

There are two methods of passing information -- by value and by reference. When a parameter is passed by value, the actual parameter is evaluated and the resulting actual value is passed. When a parameter is passed by reference, its address is passed. In the procedure declaration, a parameter that is passed by reference is preceded by "VAR."

In/Out

The third column tells you whether a parameter is an input parameter (one that passes information to the procedure) or an output parameter (one that passes information from the procedure to your program). Some parameters are both input and output parameters; in such a case, you use the parameter to pass a value to the procedure, and the procedure returns a different value to your program. All output parameters are passed by reference, and all input parameters are passed by value. Parameters that are both input and output are passed by reference.

Data Type

The fourth column contains a parameter's data type. Because these procedures are shown in Pascal-86, the data types are Pascal data types. If you are accessing the procedures from some other language, e.g., PL/M-86, you must match the parameters with variables in your program that have compatible data types. To determine what data types in your language correspond to the procedure types in column 4, see your language manual. Table 2-1 lists Pascal data types and, since some of you program in PL/M-86. shows corresponding data types in PL/M-86.

Table 2-1. Data Types for ISAM Procedure Parameters

Pascal Type	Pascal Description	PL/M-86 Type
Boolean	Simple ordinal. Predefined values are FALSE (0) and TRUE (1).	Byte
Integer	Simple ordinal of two bytes in the range -32768 through +32767.	Integer
LongInt	Simple ordinal of four bytes in the range -2,147,483,647 through +2,147,483,647. Signed.	DWord
Word	Simple ordinal of two bytes. Integers in the range 0 through 65535. Unsigned.	Word
Byte	An enumerated type defined (0 .. 255).	Byte
StringPtr	A pointer to a record type defined: String = Record Len:Integer; Max:Integer; Dummy:Byte Chars:ARRAY [1..maxInt] of Char; END; Byte = 0 .. 255; StringPtr = ^String; A StringPtr is a dynamic structure allocated by the NewString procedure and disposed of by the FreeString procedure described in the Common Code Manual). The calling program must never change the Max element as it is the maximum amount of memory allocated to the string.	Pointer

CHAPTER 3: ISAM FILE MANAGER PROCEDURES

This chapter lists the ISAM File Manager Procedures in alphabetical order. Use it to look up the details of a procedure as you construct a call.

ISAMAppendRecord

```
PROCEDURE ISAMAppendRecord (fileNumber: Integer;
                            newRecord: StringPtr;
                            VAR recordID: LongInt;
                            VAR status: Word);
```

OPERATION

Inserts a new record at the end of the open data file you identify and returns the I.D. number of the new record. All key files associated with the data file must also be open so that they can be updated. The record appended, pointed to by newRecord, must have the correct number of columns for the data file (the record consists of character data separated into columns by TAB characters and a final CARRIAGE-RETURN/LINE-FEED). Successive calls to ISAMAppendRecord load a new data file.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
newRecord	Value	Input	StringPtr
recordID	Reference	Output	LongInt
status	Reference	Output	Word

ISAMAttach

```
FUNCTION ISAMAttach (fileName: StringPtr;
                     password: StringPtr;
                     VAR Status: Word):
                     Integer;
```

OPERATION

Makes a previously created data file and its key files available to your program. You must provide the proper password. You must attach a file before you can open it.

FUNCTION RETURN

Returns the file number (data type Integer) assigned by the system. Associate a variable with the procedure name, then use it in other calls to identify the data file.

Parameter	Pass Method	In/Out	Data Type
fileName	Value	Input	StringPtr
password	Value	Input	StringPtr
status	Reference	Output	Word

NOTE: If the file does not have a password, the password value should be zero (0).

ISAMChangeColumns

```
PROCEDURE ISAMChangeColumns (fileNumber: Integer;
                             fromColumn: Integer;
                             toColumn : Integer;
                             columns: Integer;
                             eraseOrignal: Boolean;
                             VAR status: Word);
```

OPERATION

Inserts, deletes, duplicates or moves columns in the open data file you identify. To delete columns, place a negative number in columns; that number of columns will be deleted beginning at the column identified by toColumn. To insert columns, place a positive number in columns and zero in fromColumn; the specified number of columns will be inserted beginning at the column specified in toColumn.

To duplicate columns, place a positive number in columns and the column number of the first source column in fromColumn; the number of columns specified will be duplicated beginning at the column specified by toColumn. The setup to move columns is identical to duplicate except that you set eraseOriginal true; the number of columns specified will be moved to the column specified by toColumn and deleted at fromColumn.

When you insert or delete a column, the ISAM File Manager updates every record in the file and compresses the file. Since characers are added or deleted from each record, record I.D.'s obtained previously are no longer valid.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
fromColumn	Value	Input	Integer
toColumn	Value	Input	Integer
columns	Value	Input	Integer
eraseOriginal	Value	Input	Boolean
Status	Reference	Output	Word

ISAMClose

```
PROCEDURE ISAMClose (fileNumber: Integer
                     VAR status: Word);
```

OPERATION

Closes and deallocates memory for the open data file that you identify. This procedure does not close the key files associated with the data file.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
status	Reference	Output	Word

ISAMCloseIndex

```
PROCEDURE ISAMCloseIndex (fileNumber: Integer;
                          columnNumber: Integer;
                          VAR status: Word);
```

OPERATION

Closes and deallocates memory for the open key file associated with the column and data file you identify.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
columnNumber	Value	Input	Integer
status	Reference	Output	Word

ISAMCompress

```
PROCEDURE ISAMCompress (fileNumber: Integer;
                        VAR status: Word);
```

OPERATION

Physically removes all the records that are marked deleted, compressing the data file you identify.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
status	Reference	Output	Word

ISAMCreate

```
FUNCTION ISAMCreate (fileName: StringPtr;
                     password: Stringptr;
                     numberColumns: Integer;
                     VAR status: Word): Integer;
```

OPERATION

Creates a file with the name you specify in fileName, the password you specify in password, and the number of columns you specify in numberColumns. If a file of the same name and password exists already, it is deleted.

FUNCTION RETURN

Returns the file number, a type Integer, that the system assigns. Associate a variable with the procedure name, then use it in other calls to identify the data file.

Parameter	Pass Method	In/Out	Data Type
filename	Value	Input	StringPtr
password	Value	Input	StringPtr
numberColumns	Value	Input	Integer
status	Reference	Output	Word

ISAMCreateIndex

```
PROCEDURE ISAMCreateIndex (fileNumber: Integer;
                           columnNumber: Integer;
                           keysAreUnique: Boolean;
                           format: Byte;
                           VAR status: Word);
```

OPERATION

Creates an alphabetically or numerically sorted file for the key column and data file you specify in columnNumber and fileNumber. Set format to 0 for alphanumeric keys or to 1 for numeric keys. You may ask that the keys be unique (TRUE) or not unique (FALSE). If you ask that keys be unique and there already are nonunique values in the column, an error is returned and the key file is not created.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
columnNumber	Value	Input	Integer
keysAreUnique	Value	Input	Boolean
format	Value	Input	Byte
status	Reference	Output	Word

ISAMDecodeError

```
FUNCTION ISAMDecodeError (status: Word): StringPtr;
```

OPERATION

Locates the message associated with an ISAM error code or an operating system error code. ISAM error codes, in the range 2500 - 2514, are described in Appendix A.

FUNCTION RETURN

Returns a pointer to a string containing the message.

Parameter	Pass Method	In/Out	Data Type
status	Value	Input	Word

ISAMDeleteIndex

```
PROCEDURE ISAMDeleteIndex (fileNumber: Integer;
                           columnNumber: Integer;
                           VAR status: Word);
```

OPERATION

Deletes the sorted key file previously created for the identified column and data file. The data file must be attached, but it need not be open.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
columnNumber	Value	Input	Integer
status	Reference	Output	Wor

ISAMDeleteRecord

```
PROCEDURE ISAMDeleteRecord (fileNumber: Integer;
                            recordID: LongInt;
                            VAR status: Word);
```

OPERATION

Marks a record (row) in the identified open data file as deleted. All keys associated with the data cells in this row are also marked deleted. (Data is not physically removed until the data file is compressed.)

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
recordID	Value	Input	LongInt
status	Reference	Output	Word

ISAMDetach

```
PROCEDURE ISAMDetach (fileNumber: Integer;
                      VAR status: Word);
```

OPERATION

Makes the previously attached data file and its associated key files unavailable to your program. In addition, ISAMDetach closes the data file and closes and detaches the key files.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
status	Reference	Output	Word

ISAMExit

```
PROCEDURE ISAMExit;
```

OPERATION

Closes and detaches all attached data files and key files and exits your program. Since this procedure does not return to your program, it should be the last one you call.

ISAMFind

```
PROCEDURE ISAMFind (fileNumber: Integer
                    columnNumber: Integer;
                    valueToFind: StringPtr;
                    VAR unique: UniqueType*;
                    format: Byte;
                    VAR recordID: LongInt;
                    VAR status: Word);
```

OPERATION

Finds the first record with a value in the data file and column you identify and returns the I.D. of the record. If the column is a key column, the Unique parameter returns a 0 if the next key is a duplicate, a 1 if it is not. When the column is not a key column, Unique returns a 2. When the column is not a key column, specify an alphabetic (0) or numeric (1) search using the format parameter.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
columnNumber	Value	Input	Integer
valueToFind	Value	Input	StringPtr
unique	Reference	Output	UniqueType*
format	Value	Input	Byte
recordID	Reference	Output	LongInt
status	Reference	Output	Word

* One-byte enumerated type defined (KeyIsUnique, KeyIsNotUnique, Unknown).

ISAMFindGeneric

```
FUNCTION ISAMFindGeneric (fileNumber: Integer;
                          columnNumber: Integer;
                          valueToFind: StringPtr;
                          VAR unique: UniqueType*;
                          VAR recordID: LongInt;
                          VAR status: Word): StringPtr;
```

OPERATION

Finds the value in a key column that is greater than or equal to a value you specify in the valueToFind parameter. Returns the I.D. of the record containing the value. Unique returns 0 if the next key is a duplicate, 1 if it is not. The column you identify must be a key column.

FUNCTION RETURN

Returns a pointer to the string containing the key value found. You must subsequently free the string by calling FreeString (see the Common Code Reference Manual).

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
columnNumber	Value	Input	Integer
valueToFind	Value	Input	StringPtr
unique	Reference	Output	UniqueType*
recordID	Reference	Output	LongInt
status	Reference	Output	Word

* A one-byte enumerated type defined to be (KeyIsUnique, KeyIsNotUnique, Unknown).

ISAMFlushAllBuffers

```
FUNCTION ISAMFlushAllBuffers (conn: Word;
                              VAR status: Word): StringPtr;
```

OPERATION

Writes the contents of all buffers in memory (allocated with ISAMOpen and ISAMOpenIndex) currently assigned to a file out to the file residing on a device-subject. It can thus be thought of as a precautionary call that lets you save the contents of file buffers without going through a "close-open" sequence.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
status	Reference	Output	Word

ISAMFindNext

```
FUNCTION ISAMFindNext (fileNumber: Integer;
                       columnNumber: Integer;
                       VAR unique: UniqueType*;
                       VAR recordID: LongInt;
                       VAR status: Integer): StringPtr;
```

OPERATION

Finds the next physical record in the data file or, if the column identified in columnNumber is a key column, the next record in the sorted key file. The "next" record is related to the file pointer. Each data file and key file has an indepenent file pointer which, when the file is first opened or after it is updated or after a call to ISAMResetFile, points to the first record.

RecordID returns the I.D. of the record found. If the column is a key column, unique returns 0 if the next key is a duplicate, 1 if it is not. When the column is not a key column, unique returns 2.

FUNCTION RETURN

Returns a pointer to the string containing the content of the column in the record found.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
columnNumber	Value	Input	Integer
unique	Reference	Output	UniqueType*
recordID	Reference	Output	LongInt
status	Reference	Output	Word

* A one-byte enumerated type defined to be (KeyIsUnique, KeyIsNotUnique, Unknown).

ISAMGetStatus

```
PROCEDURE ISAMGetStatus (fileNumber: Integer;
                         VAR info: ISAMInfoType;*
                         len: Integer;
                         VAR status: Word);
```

OPERATION

Counts the number of key columns, the number of total columns, and the number of deleted or replaced records since the last compression. This information is returned in the info parameter. Specify the number of bytes to obtain from info in len.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
info	Reference	Input/	ISAMInfoType*
len	Value	Input	Integer
status	Reference	Output	Word

```
*Defined to be Record
               numIndexes: Byte;
               numcols: Integer;
               numDeletedRecords: Integer;
               END;
```

ISAMOpen

```
PROCEDURE ISAMOpen (fileNumber: Integer;
                    VAR status: Word);
```

OPERATION

Opens and allocates memory to a previously attached data file. You must open a file before you can access it.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
status	Reference	Output	Word

ISAMOpenIndex

```
PROCEDURE ISAMOpenIndex (fileNumber: Integer;
                         columnNumber: Integer;
                         VAR status: Word);
```

OPERATION

Opens and allocates memory to a previously created key file associated with a particular data file. You must open the key file before you can use it.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
columnNumber	Value	Input	Integer
status	Reference	Output	Word

ISAMRead

```
FUNCTION ISAMRead (fileNumber: Integer;
                   recordID:  LongInt;
                   VAR status: Word): StringPtr;
```

OPERATION

Copies the record you identify with recordID from an open data file to a character string that the ISAM File Manager establishes.

FUNCTION RETURN

Returns a pointer to the string containing the data record. When you are through with the string, you should delete it by calling the procedure FreeString (see the Compass Computer Common Code Manual).

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
recordID	Value	Input	LongInt
status	Reference	Output	Word

ISAMReplaceRecord

```
PROCEDURE ISAMReplaceRecord (fileNumber: Integer;
                             newRecord: StringPtr;
                             VAR recordID: Longint;
                             VAR status: Word);
```

OPERATION

Replaces the content of a specified record in an open data file. You supply the new data in newRecord and you identify the record to be replaced in recordID. The ISAM File Manager returns the new record I.D. using the same recordID parameter (it is both input and output). Any key file associated with the data file must be open so that it can be updated.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
newRecord	Value	Input	StringPtr
recordID	Reference	Input/ Output	LongInt
status	Reference	Output	Word

ISAMResetFile

```
PROCEDURE ISAMResetFile (fileNumber: Integer;
                         columnNumber: Integer;
                         VAR status: Word);
```

OPERATION

Moves the file pointer to the first record in a data file or a key file. When columnNumber is 0 or any other column number that does not identify a key column, the data file's pointer is reset to the beginning of the data file (the next record found by ISAMFindNext will be the first record of the data file). When columnNumber identifies a key column, the pointer for that key file is reset to the beginning of the key file.

Parameter	Pass Method	In/Out	Data Type
fileNumber	Value	Input	Integer
columnNumber	Value	Input	Integer
status	Reference	Output	Word

APPENDIX A: STATUS PARAMETER RETURNS

The status parameter, the last parameter for all ISAM procedures except ISAMExit, returns information on the successful or unsuccessful completion of the procedure's work.

In your call, provide a status variable of the data type Word to accept the error number (status code) passed by reference to your program. You may want to write code to check status and take appropriate action. Additionally, you can call the procedure ISAMDecodeError to return the message associated with the error number.

If status is 0, the procedure completed sucessfully. If you receive any other number, something has gone wrong. ISAM File Manager errors are listed in the table below. Operating system errors, outside of the range 2500 - 2515, are listed in the Operating System Reference Manual.

Table A-1. Status Codes and Their Messages

Status Code	Message	What to Do <<to be added>>
2500	Too many attached ISAM files	
2501	ISAM file is not attached	
2502	ISAM file is not open	
2503	ISAM file number is invalid	
2504	ISAM file is already attached	
2505	ISAM record format is invalid	
2506	ISAM Key file already exists	
2507	ISAM Key file doesn't exist	
2508	ISAM duplicate keys are invalid	
2509	ISAM key file is invalid	
2510	ISAM record does not exist	
2511	ISAM record was not found	
2512	ISAM record was deleted	
2513	ISAM descriptor file is invalid	
2514	ISAM column number is invalid	
2515	ISAM file type is invalid	

APPENDIX B: RUNNING ISAM PROGRAMS

This appendix names the subsystem file containing ISAM run-time software, the file that must be loaded in order to run a program that uses the ISAM File Manager.

It also provides names two include files, one which must be included if your calling program is written in Pascal-86 and one which must be included if your program is written in PL/M-86. The include files contain the declarations of data structures, functions, and procedures. To use the ISAM File Manager, your application must refer to the appropriate include file with the $INCLUDE statement.

> WARNING: Because a declaration in this manual may not always be up to date with respect to the most current software, you must use the declaration from the include file over the declaration from the manual if they disagree.

ISAM File Manager Software: ISAM.Common

Pascal-86 include file: ISAM.PAS.INC

PL/M-86 include file: ISAM.PLM.INC

For complete programmer instructions, including directions for loading, linking, and compiling, see the Compass Computer Program Development Guide and the appropriate language manual.