

# **FOCUS for S/390**

ADABAS Interface User's Manual & Installation Guide  
Version 7.0

# Contents

---

---

1.	Introduction .....	1-1
1.1	Operating Environments .....	1-2
1.2	Security .....	1-2
2.	Getting Started .....	2-1
2.1	Getting Started Under MVS .....	2-1
2.1.1	Interactive Access From TSO .....	2-2
2.1.2	Interactive Access From MSO .....	2-3
2.1.3	Batch Access .....	2-4
2.2	Getting Started Under CMS .....	2-5
2.2.1	Interactive Access From CMS .....	2-5
2.2.2	File Descriptions .....	2-6
3.	ADABAS Overview and Mapping Considerations .....	3-1
3.1	Overview of ADABAS .....	3-1
3.2	ADABAS Files .....	3-3
3.2.1	Inverted Lists: The ADABAS Key Structure .....	3-3
3.2.2	Field Definition Tables .....	3-4
3.2.3	Managing Data Storage With Null-suppression .....	3-6
3.3	Mapping ADABAS Structures in the Master File .....	3-7
3.3.1	ADABAS Concepts in FOCUS Terminology .....	3-8
3.3.2	ADABAS Descriptors .....	3-9
3.3.3	ADABAS Record Lengths .....	3-10
4.	The AUTOADBS Facility .....	4-1
4.1	Ease of Use Features .....	4-2
4.2	Documentation in the Master and Access Files .....	4-2
4.3	How to Use the AUTOADBS Facility .....	4-3
4.3.1	Starting the AUTOADBS Facility .....	4-4
4.3.2	The Main Menu .....	4-4
4.3.3	Generating the Master and Access Files .....	4-9
4.4	Background Execution .....	4-20

# Contents

4.5	The Generated Descriptions .....	4-22
4.5.1	Master File and Segment Attributes .....	4-22
4.5.2	Access File Attributes.....	4-23
4.5.3	Master File Field Attributes.....	4-23
4.5.4	Fieldnames.....	4-25
4.5.5	USAGE and ACTUAL Formats.....	4-27
4.5.6	Changes to the Generated Descriptions.....	4-28
4.5.7	Creating a Record of Generated Master and Access Files.....	4-29
4.5.8	Search Order for Parameter Log File in MVS.....	4-30
5.	Describing ADABAS Files to FOCUS .....	5-1
5.1	Master Files .....	5-2
5.1.1	File Attributes .....	5-3
5.1.2	Segment Attributes .....	5-4
5.1.3	Field Attributes .....	5-6
5.1.4	Specifying INDEX=I.....	5-10
5.1.5	Describing Superdescriptors.....	5-11
5.1.6	Describing Multi-value Fields and Periodic Groups With the OCCURS Attribute .....	5-14
5.1.7	Describing Multi-value Fields Within Periodic Groups .....	5-16
5.1.8	Specifying OCCURS Segment Sequence: The ORDER Field .....	5-17
5.2	Access Files.....	5-18
5.2.1	Release Declaration .....	5-19
5.2.2	Segment Attributes .....	5-21
5.2.3	Using the GROUP Attribute to Cross-reference Files.....	5-26
5.2.4	Describing Descriptors in the Access File.....	5-27
5.2.5	Implementing Embedded JOINS: KEYFLD and IXFLD .....	5-29
6.	Reporting Considerations.....	6-1
6.1	File Navigation Techniques .....	6-1
6.1.1	Referencing Subtrees and Record Retrieval.....	6-1
6.1.2	FOCUS Segment Retrieval Methodology .....	6-2
6.1.3	Missing Unique Segments .....	6-2
6.1.4	Missing Non-unique Segments.....	6-2
6.2	Selection Considerations .....	6-5
6.2.1	Selection Order in the Access File .....	6-5
6.2.2	RECORDLIMIT and READLIMIT .....	6-5
6.2.3	Optimization of the FIND Call Using Non-descriptor Fields .....	6-6
6.3	Using the JOIN Command to Process Multiple Files .....	6-7
6.3.1	Multi-field JOIN and Short-to-long JOIN Capability .....	6-8
6.4	Optimization With Null-suppression for CALLTYPE=RL .....	6-8
6.5	Optimization on Group Fields.....	6-9
6.6	Test on Group Field With Numerics .....	6-10

7.	Interface Navigation.....	7-1
7.1	Entry Segment Retrieval of ADABAS Records.....	7-1
7.1.1	Read Physical Calls.....	7-3
7.1.2	Read Logical Navigation.....	7-4
7.1.3	FIND Navigation.....	7-6
7.1.4	Read Descriptor Value (L9) Direct Calls.....	7-8
7.2	Descendant Periodic Groups and Multi-value Fields.....	7-9
7.3	Descendant ADABAS Records.....	7-9
7.3.1	Read Logical Calls Using a Starting Value.....	7-10
7.3.2	Simple FIND Calls.....	7-11
7.3.3	Complex FIND Calls.....	7-12
8.	Environment Commands.....	8-1
8.1	Multifetch and Prefetch Options.....	8-1
8.1.1	Invoking Multifetch/Prefetch.....	8-2
8.2	ADABAS Dynamic Database Number.....	8-3
8.3	Overriding Default Passwords in Specific Files.....	8-5
8.4	Running in 24-bit Mode.....	8-7
8.5	Optimization of the FIND Call Using Non-descriptor Fields.....	8-7
9.	Debugging Techniques.....	9-1
9.1	Common Errors and Response Codes.....	9-2
9.2	Using Traces.....	9-2
9.2.1	Invoking the Trace Facility.....	9-7
9.2.2	Turning Off the Trace Facility.....	9-7
9.3	Verifying the MVS Environment.....	9-8
9.4	Verifying the CMS Environment.....	9-8
9.5	Verification of Installation and AUTOADBS.....	9-9
9.6	Follow-up.....	9-9

# Contents

A.	Installation Instructions .....	A-1
A.1	Pre-installation and Maintenance Requirements .....	A-1
A.1.1	Software Requirements .....	A-2
A.1.2	Memory Requirements .....	A-2
A.1.3	Maintenance .....	A-2
A.1.4	Using the DDCARD ddname to Specify Site Defaults .....	A-3
A.2	Installing the ADABAS Interface in MVS.....	A-4
A.2.1	Unloading the Distribution Tape .....	A-4
A.2.2	Allocating the Interface Load Modules .....	A-6
A.2.3	Accessing the Software AG Load Library .....	A-7
A.2.4	Preparing the Run-time Libraries .....	A-7
A.2.5	Installing in the MSO Environment (Optional).....	A-10
A.2.6	Installing AUTOADBS .....	A-10
A.2.7	Testing the ADABAS Interface Installation.....	A-15
A.3	Installing the ADABAS Interface in CMS.....	A-17
A.3.1	Preparing the ADABAS Interface Load Libraries.....	A-18
A.3.2	Creating the ADAUSER TEXT File .....	A-18
A.3.3	Accessing the ADABAS Nucleus .....	A-19
A.3.4	Installing AUTOADBS for CMS .....	A-19
A.3.5	Testing the ADABAS Interface Installation.....	A-23
B.	Interface Error Messages.....	B-1
B.1	Messages for MVS and CMS.....	B-1
C.	Sample File Descriptions .....	C-1
C.1	VEHICLES Sample Files.....	C-1
C.2	EMPLOYEEES Sample Files .....	C-4
C.3	ACUSTOMR Sample Files.....	C-8
C.4	AMKTORDR Sample Files .....	C-8
I.	Index.....	I-1

Information Builders, the Information Builders logo, and FOCUS are registered trademarks of Information Builders, Inc.

IBM and MVS are registered trademarks of International Business Machines Corporation.

ADABAS is a registered trademark of Software A.G.

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. In most, if not all cases, these designations are claimed as trademarks or registered trademarks by their respective companies. It is not this publisher's intent to use any of these names generically. The reader is therefore cautioned to investigate all claimed trademark rights before using any of these names other than to refer to the product described.

Copyright © 1998, by Information Builders, Inc. All rights reserved. This manual, or parts thereof, may not be reproduced in any form without the written permission of Information Builders, Inc.

Printed in the U.S.A.

# Preface

---

This manual describes how to install and use the ADABAS Interface, which extends the reporting facilities of FOCUS to ADABAS database structures.

To use the manual effectively, you must be familiar with standard FOCUS reporting features as described in the *FOCUS for IBM Mainframe User's Manual*. This manual also assumes you have access to the ADABAS Database Management System (DBMS) and are familiar with ADABAS processing.

This manual supersedes the *FOCUS Interface to ADABAS User's Manual* (DN1000017.0390) and incorporates information from the *Technical Memo for the ADABAS Interface Installation Guide* (TM7911.1).

You can install the ADABAS Interface in the MVS and VM/CMS operating environments. Detailed installation instructions are included in Appendix A, *Installation Instructions*.

## How This Manual Is Organized

---

This manual is organized into nine chapters and three appendices:

- Chapter 1, *Introduction*, describes how the Interface operates and how to use it.
- Chapter 2, *Getting Started*, describes the Interface execution instructions.
- Chapter 3, *ADABAS Overview and Mapping Considerations*, describes the components of an ADABAS database and how to map ADABAS structures to FOCUS.
- Chapter 4, *The AUTOADBS Facility*, describes how to automatically generate Master and Access Files for ADABAS files and user views.
- Chapter 5, *Describing ADABAS Files to FOCUS*, describes how ADABAS features relate to FOCUS and provides details on the components of FOCUS Master and Access Files.
- Chapter 6, *Reporting Considerations*, describes advanced reporting topics, such as the FOCUS SET ALL command and the JOIN command.
- Chapter 7, *Interface Navigation*, describes how the Interface navigates an ADABAS database through direct calls in read only mode.
- Chapter 8, *Environment Commands*, lists Interface environment commands that display and change the parameters that govern your FOCUS session.
- Chapter 9, *Debugging Techniques*, describes installation verification procedures and the tracing facilities for the Interface.

## Preface

- Appendix A, *Installation Instructions*, describes how to install the Interface in the MVS and CMS operating environments. These instructions apply to FOCUS Release 7.0 and higher.
- Appendix B, *Interface Error Messages*, lists ADABAS Interface error messages.
- Appendix C, *Sample File Descriptions*, contains sample Master Files, Access Files, and Field Definition Tables for the examples included throughout this manual.

## Documentation Conventions

---

The following syntax conventions and terminology are used in this manual:

UPPERCASE	Commands and required keywords are presented in uppercase and must be typed as shown. (In some cases, a shorter unique truncation is acceptable.)
<i>lowercase</i>	User-supplied parameters are presented in lowercase italic.
Punctuation	Required as shown.
—	Underscore indicates a default value.
{ }	Braces enclose groups of required parameters; select one.
[ ]	Brackets enclose optional parameters; none is required.
...	Horizontal ellipses indicate a continuation of syntax.
. . .	Vertical ellipses indicate intervening commands for syntax.
Master File	An updated term used throughout this manual instead of “Master File Description (MFD).”
Access File	An updated term used throughout this manual instead of “Access File Description (AFD).”

### Note:

- At the command level, FOCUS accepts syntax in mixed case, uppercase, or lowercase but transmits it in uppercase.
- In sample sessions, FOCUS, ADABAS Interface, or system responses are represented in uppercase; user responses are presented in lowercase.

## Related Publications

---

Related publications include:

- *FOCUS for IBM Mainframe User's Manual Release 7.0* (DN1000983.0495).
- *FOCUS for IBM Mainframe MVS/TSO Installation Guide Release 7.0* (DN1000994.0897).
- *FOCUS for IBM Mainframe CMS Installation Guide Release 7.0* (DN1000993.0196).
- *FOCUS for IBM Mainframe Multi-Session Option Installation and Technical Reference Guide Release 7.0* (DN1000966.1095).
- *FOCUS Summary of New Features CD Release 7.0* (DN1001037.0597).

**Note:** The title and Document Number (DN) information provided here are accurate as of this printing. To ensure up-to-date information when ordering, please consult the latest *Information Builders Publications Catalog*.

You can also visit our World Wide Web site, <http://www.ibi.com>, to view a current listing of our publications and to place an order.

## Information Builders Systems Journal

---

The *Information Builders Systems Journal* is a unique technical publication dedicated to providing you with the latest information to enhance your use of all Information Builders products.

Through its detailed articles, illustrated with code, screen shots, and other visuals, the Journal challenges you to develop better reporting habits, customize features to enhance your systems applications, use its tips and techniques for better performance and productivity, and so much more.

You can order the *Information Builders Systems Journal* from the *Publications Catalog* or from our World Wide Web site, <http://www.ibi.com>.

## Customer Support

---

Questions about the ADABAS Interface or FOCUS?

Call Information Builders Customer Support Service (CSS) at (800) 736-6130 or (212) 736-6130. Customer Support Consultants are available between 8:00 a.m. and 8:00 p.m. EST to address all your ADABAS Interface and FOCUS questions. Information Builders consultants can also give you general guidance regarding product capabilities and documentation. Please be ready to provide your six-digit site code number (xxxx.xx) when you call.

You can also access support resources online, 24 hours a day, through our World Wide Web site, <http://www.ibi.com>. To learn about the full range of available support services, ask your Information Builders representative about InfoResponse.

You can upload your questions to the FOCWIZARD or FOCSERVICES forum on CompuServe. We do not accept questions via FAX.

## Questions About the ADABAS Interface

---

See the section on follow-up in Chapter 9, *Debugging Techniques*, for specific information you will need to supply when you have questions about the ADABAS Interface.

## Questions About FOCUS

---

To help our consultants answer your FOCUS queries most effectively, be ready to provide the following information:

- The FOCEXEC procedure (preferably with line numbers).
- Master File with picture (provided by CHECK FILE).
- Access File.

- Run sheet (beginning at login, including call to FOCUS), containing the following information:
  - ? RELEASE
  - ? FDT
  - ? LET
  - ? LOAD
  - ? COMBINE
  - ? JOIN
  - ? DEFINE
  - ? STAT
  - ? SET/? SET GRAPH
  - ? USE
  - ? TSO DDNAME or CMS FILEDEF
- The exact nature of the problem: for example, are the results or is the format incorrect; does an abend occur; are the text or calculations missing or misplaced; is this problem related to any other problem?
- Has the procedure ever worked in its present form? Has it been changed recently?
- What release of the operating system are you using? Has it, FOCUS, or an Interface system changed?
- Is this problem repeatable?

## **Information Builders Consulting and Training**

---

Interested in training? Information Builders Education Department offers a wide variety of training courses for this and other Information Builders products. For more information, course descriptions, locations, and dates, or to register for classes, visit our World Wide Web site (<http://www.ibi.com>) or call (800) 969-4636 to speak to an Education Representative.

## **User Feedback**

---

In an effort to produce effective documentation, the Documentation Services staff at Information Builders welcomes any opinion you can offer regarding this manual. Please use the Reader Comments form at the end of this manual to relay suggestions for improving the publication or to alert us to corrections.

Thank you, in advance, for your comments.

# 1 Introduction

## Topics:

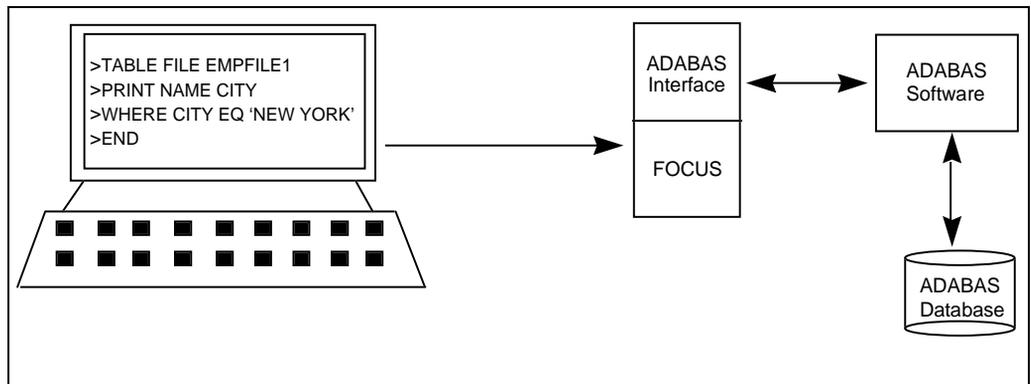
- Operating Environments
- Security

The ADABAS Interface enables you to access ADABAS databases using FOCUS. The Interface supports all FOCUS features for read access and can be purchased with the standard FOCUS product.

The primary function of the Interface is to extract data from an ADABAS structure and pass it to FOCUS for reporting purposes. The Interface issues the instructions that ADABAS uses to retrieve information from a database. Since this process is transparent, you do not need to learn ADABAS commands.

The Interface does not recreate the data in FOCUS format. Instead, it links the FOCUS report facility and the ADABAS file using FOCUS Master and Access Files that you create based on your ADABAS database. FOCUS uses Master and Access Files to understand the data. The Master File describes the field layout and is the logical description of the file. The Access File describes the physical attributes of the file, such as the database number, file number, descriptors, security, and so on. Figure 1-1 illustrates how the ADABAS Interface works.

Write access is not supported, so you cannot change an ADABAS structure using the Interface. For example, since the MODIFY and FSCAN facilities of FOCUS are maintenance and editing facilities, you cannot use them with the Interface. Additionally, the Interface does not support the alternate file view feature of FOCUS.



*Figure 1-1. ADABAS Interface Overview*

## 1.1 Operating Environments

---

The Interface is available in the following environments:

- MVS/TSO (batch and online)
- FOCUS Multi-Session Option (MSO)
- VM/CMS

It is compatible with ADABAS Release 3.2.1 and higher. The features in this manual are fully supported for FOCUS Release 7.0 and higher.

## 1.2 Security

---

The integrity of ADABAS data is not jeopardized by the Interface because FOCUS uses standard read-only ADABAS calls.

Your operating system's security applies to the Interface. ADABAS security also automatically applies to the Interface.

The Interface enables you to read password-protected ADABAS files by supplying the password in the Access File. This password can also be set dynamically in a session so that one Access File can be shared by many users (each with a different password). For more information, see the SET PASSWORD command discussed in Chapter 8, *Environment Commands*.

Additionally, you can implement the security features available with FOCUS to restrict users from accessing ADABAS databases.

FOCUS database administration enables you to do the following:

- Limit the number of users who have access to specific files.
- Restrict users from reading specific files, segments, fields, or records for certain field values.

For more information on FOCUS database administration, see the *FOCUS for IBM Mainframe User's Manual*.

## 2 Getting Started

---

**Topics:**

- Getting Started Under MVS
- Getting Started Under CMS

The following sections contain the commands necessary to run the ADABAS Interface in the following environments:

- MVS in the TSO environment, both interactive and in batch mode.
- MVS in the MSO environment.
- CMS environment.

### 2.1 Getting Started Under MVS

---

Interactive access from TSO and MSO and batch access are explained in the following sections.

## 2.1.1 Interactive Access From TSO

---

Following is a sample CLIST that, upon execution, will allocate the files necessary to run the Interface interactively.

```

PROC 0
CONTROL NOMSG NOLIST
WRITE *****
WRITE *    WELCOME TO THE IBI ADABAS INTERFACE!
WRITE *****
/*
ALLOC F(FOCEXEC) DA('userid.FOCEXEC.DATA') SHR REUSE
ALLOC F(MASTER) DA('userid.MASTER.DATA') SHR REUSE
ALLOC F(FOCADBS) DA('userid.FOCADBS.DATA') SHR REUSE
ALLOC F(USERLIB) DA('highlvl.ADABAS.LOAD' -
                    'highlvl.FUSELIB.LOAD') SHR REUSE
ALLOC F(FOCLIB) DA('highlvl.FOCLIB.LOAD') SHR REUSE
ALLOC F(ERRORS) DA('highlvl.ERRORS.DATA' -
                  'highlvl.ADABAS.DATA') SHR REUSE
ALLOC F(DDCARD) DA('SOFTWARE.AG.ADARUN') SHR REUSE
/* ALLOC F(FSTRACE) DA(*)
/* ALLOC F(FSTRACE) DA('userid.FSTRACE') MOD REUSE
/* ALLOC F(FSTRACE4) DA('userid.FSTRACE4') MOD REUSE
/* ALLOC F(FSTRACE5) DA('userid.FSTRACE5') MOD REUSE
ALLOC F(SYSOUT) DA('userid.SYSOUT') MOD REUSE
ALLOC F(SYSPRINT) DA('userid.SYSPRINT') MOD REUSE
/*
CALL 'highlvl.FOCLIB.LOAD(FOCUS)'

```

where:

*userid*                    Is the high-level qualifier for your private version of a library.

*highlvl*                Is the high-level qualifier for your site's FOCUS production libraries.

**Note:**

- To use this CLIST, the SOFTWARE AG ADABAS load library must be defined to the STEPLIB. You should define this load library in the logon procedure or LINKLIST.
- The sample CLIST assumes that you installed the ADABAS Interface using the STEPLIB allocation described in the installation procedure. If you copied the ADABAS Interface module ADBSINX into the FOCLIB.LOAD library, you should not assign the ADABAS.LOAD library to ddname USERLIB. See Appendix A, *Installation Instructions*, for detailed information about allocating the Interface load modules.
- The allocations for the FSTRACE files in this CLIST are commented out. To avoid performance problems, only allocate these files for debugging purposes.
- The first FSTRACE example, allocated to DA (\*), will print to the screen. Using the RECORDLIMIT keyword is recommended with this allocation.

For more information regarding parameters for the DDCARD ddname, see your Software AG documentation.

## 2.1.2 Interactive Access From MSO

---

MSO, a FOCUS product, is an MVS address space that supports multiple interactive users. You can access this environment from a VTAM logon or a CICS transaction.

Following is an example of JCL code that, upon execution, will allocate the files necessary to run the Interface under the MSO server.

```
//MSO      EXEC PGM=SSCTL
//STEPLIB DD DSN=highlvl.FOCLIB.LOAD,DISP=SHR
//         DD DSN=highlvl.FUSELIB.LOAD,DISP=SHR
//         DD DSN=highlvl.ADABAS.LOAD,DISP=SHR
//         DD DSN=SOFTWARE.AG.LOAD,DISP=SHR
//ERRORS  DD DSN=highlvl.ERRORS.DATA,DISP=SHR
//         DD DSN=highlvl.ADABAS.DATA,DISP=SHR
//MSGGET  DD DSN=highlvl.MSO.MSOGET,DISP=SHR      *TSO or CICS
//MSPUT   DD DSN=highlvl.MSO.MSOPUT,DISP=SHR      *Communication files
//MSOPROF DD DSN=highlvl.MSOPROF.DATA,DISP=SHR    *MSO Profile FOCEXEC
//FOCEXEC DD DSN=userid.FOCEXEC.DATA,DISP=SHR
//FOCMSO  DD DSN=highlvl.MSO.MSOCONFIG,DISP=SHR  *MSO config file
//MASTER  DD DSN=userid.MASTER.DATA,DISP=SHR
//ACCESS  DD DSN=userid.ACCESS.DATA,DISP=SHR
//FOCADBS DD DSN=userid.FOCADBS.DATA,DISP=SHR
//MSOPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//
```

where:

*highlvl* Is the high-level qualifier for your site's FOCUS production libraries.

*userid* Is the high-level qualifier for your private version of a library.

### Note:

- The DDCARD ddname is not used under MSO. The SVC number under MSO is read from the FADALINK load module, which is created by the GENFADL installation step. See Appendix A, *Installation Instructions*, for details on GENFADL.
- In the sample, ddname MSOPROF is allocated to the file that contains your MSO profile FOCEXEC. If you store this profile as a member of your *userid.FOCEXEC.DATA* library, allocate ddname MSOPROF to that library.

If you use other FOCUS Interfaces in addition to the ADABAS Interface, you must allocate additional datasets (such as load libraries, Access File libraries, and error datasets) specific to those Interfaces. For more information, see the *FOCUS for IBM Mainframe Multi-Session Option Installation and Technical Reference Guide*.

## 2.1.3 Batch Access

---

Following is an example of JCL code that, upon execution, will allocate the files necessary to run the Interface in batch mode.

```
//valid jobcard
//*
//STEP1 EXEC PGM=FOCUS
//STEPLIB DD DSN=SOFTWARE.AG.LOAD,DISP=SHR
// DD DSN=highlvl.FOCLIB.LOAD,DISP=SHR
// DD DSN=highlvl.ADABAS.LOAD,DISP=SHR
// DD DSN=highlvl.FUSELIB.LOAD,DISP=SHR
//ERRORS DD DSN=highlvl.ERRORS.DATA,DISP=SHR
// DD DSN=highlvl.ADABAS.DATA,DISP=SHR
//DDCARD DD *
ADARUN PROG=USER,MODE=MULTI,DBID=001,SVC=241,DE=3390
//MASTER DD DSN=userid.MASTER.DATA,DISP=SHR
//FOCADBS DD DSN=userid.FOCADBS.DATA,DISP=SHR
//FOCEXEC DD DSN=userid.FOCEXEC.DATA,DISP=SHR
/*FSTRACE4 DD DSN=userid.FSTRACE4,DISP=SHR
/*FSTRACE5 DD DSN=userid.FSTRACE5,DISP=SHR
/*FSTRACE DD DSN=userid.TEST.FSTRACE,DISP=(,CATLG,DELETE),
//* SPACE=(TRK,(50,10),RLSE),UNIT=SYSDA
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
EX NEWTEST
FIN
/*
```

where:

*highlvl* Is the high-level qualifier for your site's FOCUS production libraries.

*userid* Is the high-level qualifier for your private version of a library.

DDCARD ddname can be a dataset or allocated instream. See your Software AG documentation for more information about DDCARD ddname.

The ddnames MASTER, FOCADBS, and FOCEXEC allow concatenated datasets. For example:

```
//MASTER DD DSN=userid.MASTER.DATA,DISP=SHR
// DD DSN=userid.ADABAS.MASTER.DATA,DISP=SHR
// DD DSN=userid.MYMFD.MASTER.DATA,DISP=SHR
```

### Note:

- The allocations for the FSTRACE files in this JCL example are commented out. To avoid performance problems, only allocate these files for debugging purposes.
- The allocations for the FSTRACE files can be either a file that already exists (for example, see FSTRACE4) or allocated in the JCL (see the example for FSTRACE).

## 2.2 Getting Started Under CMS

---

In the CMS environment, the Interface operates as another application program making calls to ADABAS. All files, FILEDEFS, and disk workspace that FOCUS normally requires under CMS must be present.

### 2.2.1 Interactive Access From CMS

---

Below is a general outline of steps to follow when invoking the Interface. The details vary from site to site:

1. After logging on to VM, link to and access the FOCUS production disk, which should have a copy of the Interface load libraries and ADAUSER TEXT.
2. Issue the GLOBAL TXTLIB command to identify the ADABAS text library. The ADABAS text files required by the Interface must be available at runtime, because FOCUS loads the Interface dynamically from a TXTLIB. This TXTLIB must be accessible through a GLOBAL command either before entering FOCUS or from FOCUS. The syntax is

```
GLOBAL TXTLIB ADABAS textlib FUSELIB
```

where:

*textlib*                    Is the user-supplied value for the ADABAS text library.

3. Invoke FOCUS.

**Note:** In the EXEC used to call FOCUS, there should be a step to execute NUCXTNTS EXEC. This step establishes a link between a VM FOCUS user's machine and the ADABAS nucleus. It must be executed *before* invoking FOCUS.

## 2.2.2 File Descriptions

---

The Interface requires a Master and Access File for each ADABAS file referenced by FOCUS. In CMS, file descriptions and FOCEXECs exist as separate files. The naming conventions are:

<b>Filetype</b>	<b>Contents</b>
MASTER	Master Files.
FOCADBS or ACCESS	Access Files.
FOCEXEC	FOCUS procedures.

For more information about Master and Access Files, see Chapter 5, *Describing ADABAS Files to FOCUS*.

## 3 ADABAS Overview and Mapping Considerations

---

### Topics:

- Overview of ADABAS
- ADABAS Files
- Mapping ADABAS Structures in the Master File

This chapter provides an overview of ADABAS and ADABAS files. It discusses ADABAS and FOCUS concepts and explains how ADABAS file structures are mapped to FOCUS in Master and Access Files. Samples of ADABAS FDTs and FOCUS Master and Access Files can be referenced in Appendix C, *Sample File Descriptions*.

### 3.1 Overview of ADABAS

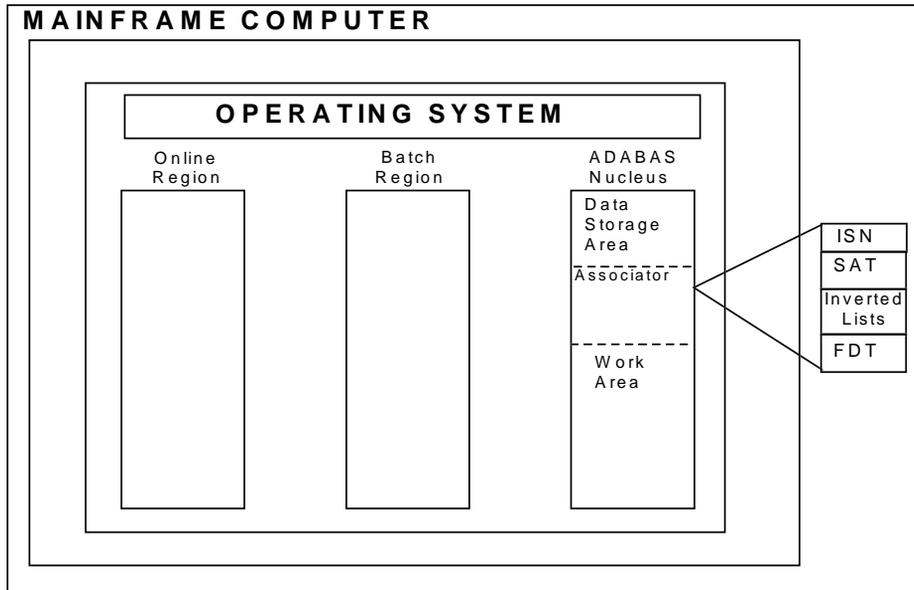
---

ADABAS is a field-oriented DBMS. Data retrievals and updates are performed on a field-by-field basis.

Since ADABAS data retrieval occurs at the field level rather than at the record level, your applications may be designed without consideration for the physical organization and maintenance of the record. Data is accessed in a variety of ways (in physical sequence, in logical sequence, or by Internal Sequence Numbers), thereby enabling you to tailor the access to address your specific needs.

## ADABAS Overview and Mapping Considerations

The ADABAS DBMS consists of several components. The following is an illustration of the ADABAS environment.



**Figure 3-1. The ADABAS Environment**

The following are the components of the ADABAS environment shown in Figure 3-1:

- Operating System** Is a set of programs (software) that control the operation of the computer (hardware).
- Online Region** Is part of the Dynamic Area of the computer. It is the section in which multiple jobs execute simultaneously. Online Region is also known as the Foreground Region (applies only to TSO).
- Batch Region** Is also part of the Dynamic Area of the computer. Jobs become a series of commands that are grouped together and processed in batches (applies only to batch jobs).
- Data Storage Area** Contains the actual data. The data is stored in compressed form.

Associator	Stores relationships about the data. It contains the following components: <ul style="list-style-type: none"> <li>• Internal Sequence Number (ISN) list, also called the Address Converter, used to determine the physical location of data.</li> <li>• Space Allocation Tables (SATs) to control storage space.</li> <li>• Inverted lists for the defined descriptors. Inverted lists and descriptors are discussed in Section 3.2.1, <i>Inverted Lists: The ADABAS Key Structure</i>.</li> <li>• Field Definition Tables (FDTs), which contain detailed data structure information for each field and descriptor.</li> </ul>
Work Area	Is a “scratch pad” used by ADABAS to build ISN lists and to sort and store records that your program requires.

The Multi-Programming Module (MPM), the cornerstone of the ADABAS Nucleus, contains the logic that enables multiple programs, both batch and online, to access ADABAS databases simultaneously.

Each database has its own data storage area, associator, and work area, in addition to its own unique database number.

## 3.2 ADABAS Files

---

When accessing ADABAS, FOCUS uses logical files as the unit of data retrieval. An ADABAS file is identified in one of two ways: by name, which can consist of up to 32 characters, or by the three-digit file number. The filename and number must be unique within each database.

All file and field documentation for the ADABAS database can be stored in the Predict dictionary and the Data Definition Module (DDM). The information about the data structure includes processing uses, file ownership, and field descriptions.

### 3.2.1 Inverted Lists: The ADABAS Key Structure

---

An ADABAS file is defined with a set of indexes called descriptors. The ADABAS term for these values, which reside in the File Associator Table, is inverted lists. The inverted lists contain the values of the descriptors, a count of the total number of records in which each value appears, and the Internal Sequence Numbers (ISNs), in ascending order, associated with each occurrence.

ADABAS utilities create and maintain an inverted list for each descriptor identified. These lists store data in either ascending (the default) or descending sequence by the value of the key. A single file may have up to 200 inverted lists associated with it.

The descriptors are identified to speed data location and to retrieve specific, frequently required data from ADABAS files. Of the available types of inverted lists, the ADABAS Interface supports the following:

<b>Inverted List</b>	<b>Description</b>
Descriptors	Key values associated with a single field (also called elementary field descriptors).
Subdescriptors	Key values associated with part of a single field.
Superdescriptors	Key values associated with all or part of two to five fields.

In the following text, “descriptor” refers to all three types of descriptors interchangeably, unless otherwise indicated.

Two or more files may be related in these ways:

- The database administrator may couple several files based on a common descriptor key in each file.
- An application program may logically relate files if there are fields in one file that can be used to access a key in another.

### 3.2.2 Field Definition Tables

---

Field description information for ADABAS files is maintained in the FDT, which is part of the ADABAS associator. ADABAS uses this information when accessing files. The FDT includes the following information:

<b>Field Information</b>	<b>Description</b>
Field Level Indicator	Denotes a hierarchical relationship between fields. In an ADABAS record description the levels are 1, 2, 3, and so on.
ADABAS Field Name	Is a two-character name used by ADABAS to identify the field. This name is unique to the file. The first character must be alphabetic, and the second character can be alphabetic or numeric. Field names E0 through E9 are reserved for internal ADABAS use.
Standard Length	Indicates the length of the field in bytes.
Standard Format	Indicates the format of the field. Refer to the table that follows for the acceptable formats and their meanings.

<b>Field Information</b>	<b>Description</b>
Field Properties	Indicates if the field value is null suppressed (NU), if the field is of fixed or variable length, and the number of field occurrences.
Descriptor Status	Indicates if the field is a descriptor, subdescriptor, superdescriptor, subfield, or superfield.
Field Type	Indicates if a field is a group (G), multi-value (MU), or periodic group (PE). Multi-value fields and periodic groups are examples of multiply occurring fields.

Typically, an ADABAS FDT contains some, but not all, of the field characteristics discussed above. To view the FDT, Software AG provides the ADAREP report. See your Software AG documentation for more information on how to run the ADAREP report.

The external field name is in the DDM (Data Definition Module from the Predict dictionary).

External Field Name	Is a 32-byte name used in a NATURAL program to identify the field. This name is unique to the file.
---------------------	---

The following table shows the acceptable standard formats for the ADABAS fields and their meanings:

<b>Format</b>	<b>Meaning</b>
A	Alphanumeric data field with a maximum of 253 bytes (126 for descriptor fields).
U	Zoned decimal data field with a maximum of 29 bytes (unsigned, unpacked data).
P	Signed packed decimal data field with a maximum of 15 bytes.
B	Unsigned binary data field with a maximum of 126 bytes.
F	Single-precision, floating point data field that is always four bytes long.
G	Decimal, double-precision integer field with an eight-byte maximum.

## ADABAS Overview and Mapping Considerations

The following column on the left illustrates a partial ADABAS FDT. The bold numbers on the left refer to the numbered annotations that follow:

PAY-FILE	
FIELD DESCRIPTION (from FDT)	FIELD NAME (from DDM)
<b>1.</b> 01, PS, 08, P, NU, DE	SSN
<b>2.</b> 01, PW, 04, P	HOURLY_WAGE
<b>3.</b> 01, MI, PE	MONTHLY_INFO
02, MH, 04, P	MONTHLY_HOURS
02, MW, 04, P	MONTHLY_WAGES
02, MT, 04, P	MONTHLY_TAX
<b>4.</b> 01, PT, 04, P, DE, MU	TAX
01, PC, 08, P, DE, MU	CHILD_SSN

- 1.** The first field, SSN, is an elementary-level field. Its internal name is PS, it is eight bytes long, and it is in packed decimal data format. The field is null suppressed, as indicated by the NU, and it is a descriptor (DE).
- 2.** HOURLY\_WAGE is also an elementary-level field. It is called PW internally, is four bytes long, and is also in packed decimal data format.
- 3.** MONTHLY\_HOURS, MONTHLY\_WAGES, and MONTHLY\_TAX are all subordinate fields of the MONTHLY\_INFO periodic group (PE). Internally, they are called MH, MW, and MT, respectively. Each is four bytes long in packed decimal data format.
- 4.** TAX and CHILD\_SSN are elementary-level fields. TAX is called PT internally, is four bytes long in packed decimal data format, and is a multi-value (MU) field with a descriptor (DE) associated with it. CHILD\_SSN is called PC internally, is eight bytes long in packed decimal format, and is also a multi-value field with a descriptor associated with it.

For more information about ADABAS periodic groups (PE) and multi-value (MU) fields, see the section entitled *ADABAS Files With Variable-length Records and Repeating Fields* in Section 3.3.3, *ADABAS Record Lengths*.

### 3.2.3 Managing Data Storage With Null-suppression

---

Null-suppression is one of the methods ADABAS employs to manage data storage. Only the fields within a record that contain data are stored. Fields containing blanks for alphabetic characters or zeros for numeric data are not stored. The fields that are not present are also transparent to you. When a FOCUS procedure accesses a record containing a null-suppressed field, ADABAS expands it to its full length and includes the null values of blanks or zeros.

When the null-suppressed field is a descriptor or part of a superdescriptor, no entry is made for the record containing that field in the inverted list. It would not be productive to have the inverted list direct you to records in which the descriptor has no data.

A field with null-suppression appears with the NU attribute in the ADABAS FDT.

### 3.3 Mapping ADABAS Structures in the Master File

---

For the ADABAS Interface to access ADABAS files, you must describe each ADABAS file you use in a Master and Access File. The logical description of an ADABAS file is stored in a Master File, which describes the field layout.

Standard Master File attributes are generally used to describe ADABAS files, but there are concepts in ADABAS file processing that require special terminology at the field and segment level.

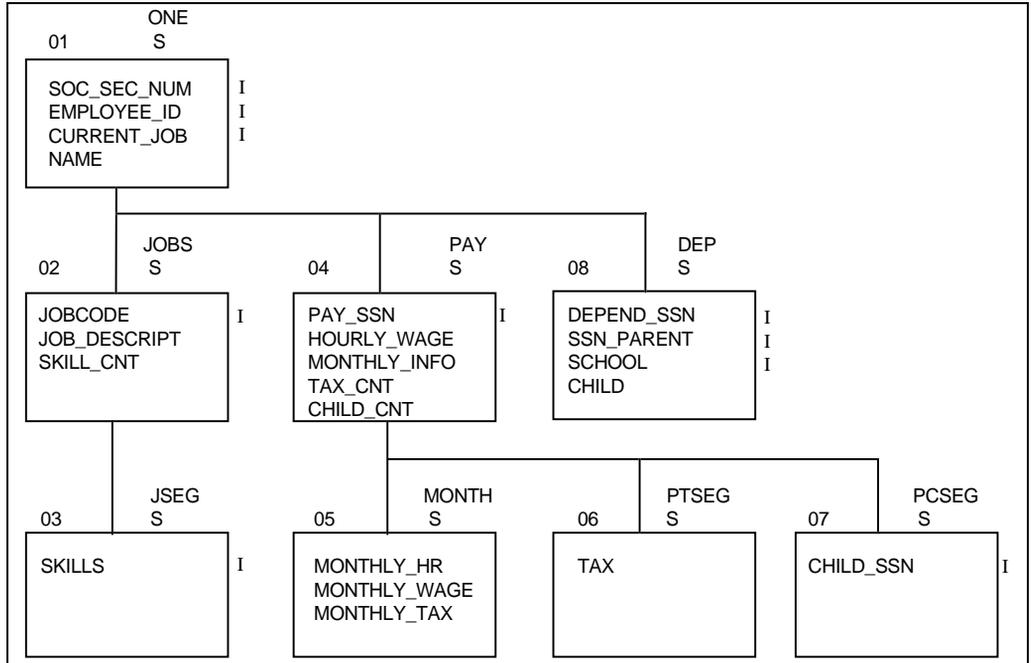
An Access File describes the physical attributes of the ADABAS file, such as the database number, file number, descriptors, and security. See Chapter 5, *Describing ADABAS Files to FOCUS*, for more information about Master and Access Files.

The preferred way to create Master and Access Files is to use the AUTOADBS facility described in Chapter 4, *The AUTOADBS Facility*. For additional information about describing external files to FOCUS, refer to the *FOCUS for IBM Mainframe User's Manual*.

### 3.3.1 ADABAS Concepts in FOCUS Terminology

FOCUS uses a non-procedural language to create reports, graphs, and extract files. It also enables you to access data files without knowing the details of the file structure or access method.

FOCUS treats any data file as either a single-path or multi-path hierarchy. Graphically, information is laid out using an inverted tree structure, as in the following sample STAFF file. In the example, the letter I to the right of a field indicates an indexed field.



The most general information appears at the top, and the more specific information appears under it. Each box in the structure is referred to as a segment. A database can consist of one or more logically related segments from a FOCUS point of view.

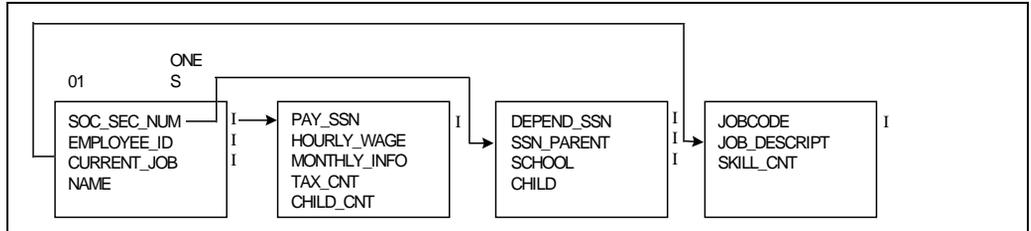
When logically related information is retrieved using this database structure, multiple occurrences of each segment are created. Each occurrence of a segment is called a segment instance. Each ADABAS database is equivalent to a collection of logically related segment instances.

The retrieval sequence of the segments is determined by the view of the structure, that is, the order of the segments from top to bottom, left to right. The segment at the top is the parent, or root, segment. The segments under the parent are the child, or descendant, segments.

Generally, parent and child segments have a one-to-many relationship; that is, a single parent has multiple occurrences of one child segment. However, FOCUS also handles one parent-one instance only of a child segment.

ADABAS uses specific concepts and techniques for organizing data. It holds data in logically distinct files that are interrelated using fields that share common formats and values called descriptors.

Within a single ADABAS file, records that contain multiple occurrences of a field can vary in length and format. The following example illustrates an ADABAS structure containing four separate files that are linked by common fields:



Many ADABAS structures are more complex than the preceding example. The following sections illustrate the mapping of typical ADABAS structures to FOCUS.

### 3.3.2 ADABAS Descriptors

Descriptors are fields, partial fields, or groups of complete and partial fields used by ADABAS to select records in a file. ADABAS descriptors correspond to indexed fields in FOCUS.

There are three types of descriptors supported by the ADABAS Interface:

- Descriptors (DE), which have a value associated with a single field.
- Superdescriptors (SPR), which have a key value associated with all or part of two to five fields (see your Software AG documentation for current SPR limitations).
- Subdescriptors (NOP), which have a key value associated with part of a single field.

All three types of descriptors are collectively referred to as “descriptors.”

## ADABAS Overview and Mapping Considerations

Consider the ADABAS FDT for the STAFF file:

<u>STAFF FILE FDT</u>	
FIELD DESCRIPTION (from FDT)	FIELD NAME (from DDM)
01, ES, 08, P, NU, DE	SOC_SEC_NUM
01, EI, 06, A, NU, DE	EMPLOYEE_ID
01, EJ, 03, A, DE	CURRENT_JOB
01, EN, 24, A	NAME

Notice that SOC\_SEC\_NUM, EMPLOYEE\_ID, and CURRENT\_JOB are labeled as descriptors. Any one of these three fields can be used to search the STAFF file.

ADABAS descriptors, superdescriptors, and subdescriptors must be declared in FOCUS Master and Access Files in certain ways:

- When creating Master Files for fields that are descriptors, use the FOCUS INDEX=I attribute. The Access File must also contain information about the descriptors, specifying the TYPE as DSC, SPR, or NOP. Identifying descriptors in the Access File directs FOCUS to inverted lists to speed retrieval of these fields.
- ADABAS fields described as superdescriptors must be declared to FOCUS in the Master File using the GROUP attribute. In the Access File, the TYPE must be SPR.

See Chapter 5, *Describing ADABAS Files to FOCUS*, for more information about declaring a descriptor, superdescriptor, and subdescriptor to FOCUS.

### 3.3.3 ADABAS Record Lengths

---

ADABAS uses files with both fixed-length and variable-length records. FOCUS, in turn, treats the different types of files in different ways. The following sections describe the differences between the two types of files.

## ADABAS Files With Fixed-length Records

---

Unless an ADABAS file has multi-value fields or periodic groups, each field occurs once in each record. When describing an ADABAS file, you do not need to describe all the fields in your FOCUS Master File, just the fields you use. Note that if you choose a periodic group, all fields in the periodic group must be defined. A single simple ADABAS file maps to a single FOCUS segment, and each field you use in the ADABAS file becomes a field in the FOCUS segment.

ADABAS RECORD		FOCUS SEGMENT	
DB	Name (in DDM)	Fieldname	Alias
ES	SOC_SEC_NUM	SOC_SEC_NUM	ES
EI	EMPLOYEE_ID	EMPLOYEE_ID	EI
EJ	CURRENT_JOB	CURRENT_JOB	EJ
EN	NAME	NAME	EN

## ADABAS Files With Variable-length Records and Repeating Fields

---

There are two types of ADABAS repeating fields:

- Multi-value (MU) fields.
- Periodic (PE) groups.

An MU field is a single field that occurs a variable number of times in a record. It appears as type MU in an ADABAS record description. ADABAS supports up to 191 occurrences of this type of field per record.

A PE group is a group of contiguous fields that occur a variable number of times in a single record. It appears as type PE in an ADABAS record description. Up to 191 occurrences per record are supported by ADABAS. These component fields are regular fields or MU fields. Describing MU fields within PE groups is discussed in detail in Chapter 5, *Describing ADABAS Files to FOCUS*.

**Note:** The ADABAS Interface supports the maximum number of occurrences of MU fields and PE groups allowed by Software AG.

## ADABAS Overview and Mapping Considerations

When an ADABAS file contains MU fields or PE groups, the repeating information occurs a variable number of times. The physical record length varies depending on how many times the repeating information actually appears.

In FOCUS, more than one segment is needed to accurately describe an ADABAS file with repeating data. The number of times the fields repeat is expressed in a counter field. Counter fields are discussed in Chapter 4, *The AUTOADBS Facility*.

The OCCURS segment attribute is used by FOCUS to describe MU and PE field types in a FOCUS Master File. Declaring a repeating field in a FOCUS Master File using the OCCURS attribute is discussed in detail in Chapter 5, *Describing ADABAS Files to FOCUS*.

Consider the sample ADABAS file VEHICLES. The FDT for the VEHICLES file is shown below.

```

*****
* FILE      2 (VEHICLES      ) *
*****
FIELD DESCRIPTION TABLE

      I      I      I      I      I      I      I
LEVEL I NAME I LENGTH I FORMAT I OPTIONS I PARENT OF I
      I      I      I      I      I      I      I
-----I-----I-----I-----I-----I-----I-----I
      I      I      I      I      I      I      I
  1 I AA I 15 I A I DE,UQ,NU I I I
  1 I AB I 4 I F I FI I I I
  1 I AC I 8 I A I DE I I I
  1 I CD I I I I I I I
  2 I AD I 20 I A I DE,NU I SUPERDE I
  2 I AE I 20 I A I NU I I I
  2 I AF I 10 I A I DE,NU I I I
  1 I AG I 2 I U I NU I SUPERDE I
  1 I AH I 1 I A I DE,FI I I I
  1 I AI I 1 I A I FI I I I
  1 I AJ I 6 I U I NU I SUPERDE I
  1 I AK I I I I I I I
  2 I AL I 3 I A I NU I I I
  2 I AM I 4 I P I MU,NU I I I
      I      I      I      I      I      I      I
-----I-----I-----I-----I-----I-----I-----I
SPECIAL DESCRIPTOR TABLE

      I      I      I      I      I      I      I      I
TYPE I NAME I LENGTH I FORMAT I OPTIONS I STRUCTURE I I
      I      I      I      I      I      I      I      I
-----I-----I-----I-----I-----I-----I-----I
      I      I      I      I      I      I      I      I
SUPER I AN I 4 I B I DE,NU I I AJ ( 5 - 6 ) I
      I      I      I      I      I      I      I      I
      I      I      I      I      I      I      I      I
SUPER I AO I 22 I A I DE,NU I I AG ( 1 - 2 ) I
      I      I      I      I      I      I      I      I
      I      I      I      I      I      I      I      I
-----I-----I-----I-----I-----I-----I-----I

```

## ADABAS Overview and Mapping Considerations

The MU field, MAINT\_COST, is a field in VEHICLES. It is allocated to a new segment, **AM0101**, required by the Interface, as shown in the Master File for a partial view of the VEHICLES file, illustrating the use of OCCURS=*occursname*:

```

$$$ CREATED BY AUTOADBS ON 12/10/97 AT 10.17.27 BY PMSMJB
FILENAME=ADACAR , SUFFIX=ADBSINX , $

$ ADABAS FILE = VEHICLES-FILE                                DICTIONARY = 6
SEGNAME=S01          , SEGTYPE=S , $
FIELD= REG_NUM      , ALIAS= AA      , A15  , A15  , INDEX=I , $
FIELD= CHASSIS_NUM  , ALIAS= AB      , I9   , I4   , $
FIELD= PERSONNEL_ID , ALIAS= AC      , A8   , A8   , INDEX=I , $
FIELD= CLASS        , ALIAS= AH      , A1   , A1   , INDEX=I , $
FIELD= LEASE_PUR     , ALIAS= AI      , A1   , A1   , $
FIELD= DATE_ACQ     , ALIAS= AJ      , P6   , Z6   , $
$GRMU = CAR_MAINTENANCE , ALIAS= AK      , A11  , A7   , $
FIELD= CURR_CODE    , ALIAS= AL      , A3   , A3   , $
FIELD= MAINT_COST_CNT , ALIAS= AMC     , I4   , I2   , $
FIELD= DAT_ACQ_DESC , ALIAS= AN      , A4   , A4   , INDEX=I , $
GROUP= MODEL_YEAR_MAKE , ALIAS= AO     , A28  , A22  , INDEX=I , $
FIELD= YEAR_S02     , ALIAS= AG      , P2   , Z2   , $
FIELD= MAKE_S02     , ALIAS= AD      , A20  , A20  , INDEX=I , $

SEGNAME=AM0101      , SEGTYPE=S , PARENT=S01 , OCCURS=AMC , $ MAX= 60
FIELD= MAINT_COST    , ALIAS= AM      , P7   , P4   , $
FIELD= AM0101_OCC    , ALIAS= ORDER   , I4   , I2   , $

```

In the original segment, the MAINT\_COST\_CNT (ALIAS=AMC) field contains the total number of occurrences of the MAINT\_COST field. The new segment, AM0101, contains the MAINT\_COST data field and the newly created AM0101\_OCC field. The AM0101\_OCC field contains the position of each occurrence.

This example includes the group field called CAR\_MAINTENANCE, which contains the MU field, MAINT\_COST. AUTOADBS comments out the group field with \$GRMU because its length is variable (depending on the number of occurrences of the MU field). FOCUS cannot report on a group with an unknown length. However, the component fields can be referenced for reporting purposes.

**Note:** The AUTOADBS facility would automatically create the new segment, AM0101, for you. Otherwise, you must create the new segment using the OCCURS segment attribute. See Chapter 5, *Describing ADABAS Files to FOCUS*, for more information about describing multi-value fields with the OCCURS attribute.

The following is the Access File for the VEHICLES file:

```

$$$ CREATED BY AUTOADBS ON 12/10/97 AT 10.17.27 BY PMSMJ
$$$ FILENAME=ADACAR,SUFFIX=ADBSINX,$
RELEASE=6,OPEN=YES,$

$ ADABAS FILE = VEHICLES-FILE                                DICTIONARY = 6
SEGNAM=S01 ,ACCESS=ADBS,FILENO=002,
      CALLTYPE=RL,SEQFIELD=PERSONNEL_ID,$
FIELD= DAT_ACO_DESC ,TYPE=NOP,$
FIELD= MODEL_YEAR_MAKE ,TYPE=SPR,$
FIELD=YEAR_S02 ,TYPE= ,NU=YES,$
FIELD=MAKE_S02 ,TYPE=DSC,NU=YES,$
SEGNAM=AM0101,ACCESS=MU ,FILENO=002,$ MAINT_COST
    
```

A new segment is created for the MU field. Note that for this new segment, ACCESS=MU.

Here is an example of the counter fields. The AMC field contains the total count of occurrences, the AM field contains the actual data, and the ORDER field contains the position of each occurrence.

(AMC)	(AM)	(ORDER)
MAINT_COST_CNT	MAINT_COST	AM0101_OCC
1	520	1
1	210	1
4	44	1
4	322	2
4	66	3
4	188	4
6	324	1
6	1103	2
6	566	3
6	755	4
6	988	5
6	1899	6
2	344	1
2	500	2

## ADABAS Overview and Mapping Considerations

In some cases, the ORDER field may describe the months of the year (1-12). If all occurrences for the month of June are needed, for example, you would print data in which the ORDER field is equal to 6 (for June).

Using the preceding sample data, here is an example of how to use the ORDER field; this request always selects the sixth occurrence of maintenance costs:

```
TABLE FILE VEHICLES
PRINT MAINT_COST
WHERE AM0101_OCC EQ 6
END
```

```
MAINT_COST
-----
      1899
```

Using the same data, this request always selects the second occurrence of maintenance costs:

```
TABLE FILE VEHICLES
PRINT MAINT_COST
WHERE AM0101_OCC EQ 2
END
```

```
MAINT_COST
-----
      322
      1103
      500
```

## 4 The AUTOADBS Facility

---

### Topics:

- Ease of Use Features
- Documentation in the Master and Access Files
- How to Use the AUTOADBS Facility
- Background Execution
- The Generated Descriptions

The AUTOADBS facility automatically generates Master and Access Files for ADABAS files and views based on information stored in the Predict dictionary and supplied by user selections. The facility requires FOCUS Release 6.8 or higher, ADABAS Release 5.0 or higher, and Predict Release 3.1.4 or higher. For installation instructions, see Appendix A, *Installation Instructions*.

AUTOADBS may be executed interactively or in the background as a batch job. Multiple files can be related in one Master File in interactive mode; background mode is limited to one file per execution. One or more versions of the Predict dictionary can be used concurrently to describe several ADABAS files (for example, Production and Test dictionaries).

The facility accurately describes the ADABAS file or view within the constraints of the ADABAS Interface. Superdescriptors, subdescriptors, periodic elements (PE groups), and multi-value (MU) fields are included. Comments are used to describe unsupported fields in the generated Master File.

AUTOADBS provides several options that enable you to customize the output. You may:

- Include the NATURAL column heading stored in the Predict dictionary in the Master File for use as the default column heading in FOCUS reports.
- Specify the starting position and length of the Predict dictionary fieldname. This feature strips off common prefixes from the fieldname or truncates the fieldname to 12 characters (for example, when downloading data using ON TABLE PCHOLD).
- Specify the FOCUS date format for NATURAL date fields.
- Create help files for FOCUS TableTalk (FOCDEFs) as a user option. The first three lines of the Predict dictionary REMARKS field are used to provide online TableTalk help.

## 4.1 Ease of Use Features

---

The AUTOADBS facility provides the following features:

- Default menu parameters are logged to disk and are recalled in subsequent interactive AUTOADBS sessions.
- An extensive online help facility is provided for all menus and screens.
- A list of ADABAS files and views described in the Predict dictionary is available online. The wildcard character, an asterisk (\*), is used to select a subset of files.
- The Master and Access Files may be edited without exiting AUTOADBS.
- The capability to produce a picture of the specified Master File is provided from the Main Menu. (It must be allocated to ddname MASTER in MVS.)
- Many ADABAS files may be described in one AUTOADBS session.
- AUTOADBS has several parameters that can be customized for site-specific situations. Refer to the AUTOADBS installation procedure in Appendix A, *Installation Instructions*, for details.
- Comments are provided in the generated Master and Access Files for additional documentation.

## 4.2 Documentation in the Master and Access Files

---

The generated Master and Access Files contain several commented entries. These comments are provided for your convenience and are not used by FOCUS. Comments may optionally be excluded from the files with a menu option. Comments are provided for:

- The user ID, date, and time of creation (included in both the Master and Access Files).
- The ADABAS filename for each file described.
- Fields not supported by the ADABAS Interface.
- The complete Predict dictionary fieldname, ADABAS format, and length. These comments are useful when truncating the fieldname.
- The maximum number of occurrences for PE and MU repeating fields.
- Duplicate fieldnames with the number of times that the field is encountered and the segments in which they are found.

- The component fields of superdescriptors that are composed of partial fields (described with the NOP type). The start and end positions of the partial fields are also included as comments.

## 4.3 How to Use the AUTOADBS Facility

---

Sufficient disk space is required for AUTOADBS to write the new file descriptions and parameter log file, if applicable, to the datasets specified on the initial input screen (in MVS) or to the disk accessed as filemode A (in CMS). In addition, sufficient disk space must be available for temporary work files. The amount of temporary disk space depends on the size of the files being described. In the CMS environment, a minimum of three available cylinders is recommended. AUTOADBS will use the temporary disk space for all temporary files that are created.

The following are requirements for MVS execution:

- The Master Files AUTOADBS, PREDDDB, and PREDEL must be members in a library allocated to the ddname MASTER.
- The Access Files PREDDDB and PREDEL must be members in a library allocated to the ddname FOCADBS.
- The AUTOADBS FOCEXEC must be a member in a library allocated to the ddname FOCEXEC. These members will be located in your site's production FOCUS libraries as a result of the Interface installation process.

For CMS execution, the files AUTOADBS FOCEXEC, AUTOADBS MASTER, PREDDDB MASTER, PREDDDB FOCADBS, PREDEL MASTER, and PREDEL FOCADBS must be available to you on an accessed minidisk. This minidisk is usually the FOCUS production disk. The files are located here as a result of the Interface installation process.

If you are not familiar with Master and Access File attributes and ADABAS data attributes, refer to Chapter 5, *Describing ADABAS Files to FOCUS*.

Before you begin, you need to determine the following:

- The Predict dictionaries that will describe the file (if your site installation supports multiple dictionary access from AUTOADBS).
- The filenames of the files to be described.
- The database number of the files to be described.
- The fields to be used as the SEQFIELD, the desired CALLTYPE, and the ADABAS security password (if one is assigned) for each file described.

## The AUTOADBS Facility

- The file that is the root segment and, if necessary, the files that are included as descendants.
- The cross-reference keys (KEYFLD and IXFLD) in each file, which will be used when describing multiple files in one view.

Contact your site's ADABAS database administrator for specific details of your files. Since you can create several Master Files in the same AUTOADBS session, you may want to prepare several file views before proceeding.

### 4.3.1 Starting the AUTOADBS Facility

---

To start AUTOADBS, enter your FOCUS session and type the following at the FOCUS command level:

```
EX AUTOADBS
```

Press the Enter key.

### 4.3.2 The Main Menu

---

The following is an example of the initial AUTOADBS screen in MVS. User entries on this and following screens are in lowercase.

```
Main Menu      Master File Generation Facility for ADABAS

Master Filename =====> sample

Describe ADABAS Files:
  File Name =====> sysdic*
  PREDICT Dictionary=>   ( )

Description will be a member of:
  Master Target PDS =====> USER1.MASTER.DATA
  Access Target PDS =====> USER1.FOCADBS.DATA
  FOCDEF Target PDS  =====> USER1.FOCDEF.DATA

Replace Existing Description?=> N Start with Fieldname Position=> 1   (1-32)
Include Fieldname Comments? ==> Y           for a Total Length of => 36 (5-36)
Use NATURAL Column Headings? => N Date Display Format=====> YYMD
Use COMMENTS for FOCDEF? =====> N Display all Userviews? =====> N
Parm File => USER1.FOCADBS.DATA

PF1=Help PF2=Restart PF3=Exit PF4=Log PF5=MFD PF6=AFD PF9=Picture PF10=List
```

The following is an example of the initial AUTOADBS screen in CMS.

```

Main Menu   Master File Generation Facility for ADABAS

      Master Filename =====> sample

Describe ADABAS Files:
      File Name =====> sysdic*
      PREDICT Dictionary=>   ( )

Replace Existing Description?=> N Start with Fieldname Position=> 1   (1-32)
Include Fieldname Comments? ==> Y   for a Total Length of =====> 36 (5-36)
Use NATURAL Column Headings? => N Date Display Format =====> YYMD
Use COMMENTS for FOCDEF? =====> N Display all Userviews? =====> N
Parm File => ADBS$PRM FOCADBS A

PF1=Help PF2=Restart PF3=Exit PF4=Log PF5=MFD PF6=AFD PF9=Picture PF10=List
    
```

**Note:** With the exception of the Main Menu, AUTOADBS screens are identical in MVS and CMS.

On the Main Menu, provide information about the ADABAS files to be described, a name for the Master and Access Files, and target datasets for the generated descriptions (MVS only). Select the appropriate options desired. After you have finished with the Main Menu, press the Enter key to begin processing.

The following options are available on the Main Menu:

- |                       |  |
|-----------------------|--|
| Master Filename       | Is a one- to eight-character name for the file description. (In MVS, this name must be a valid member name. In CMS, this name must be a valid filename.)   |
| File Name             | Is the 1- to 32-character ADABAS filename as registered in the Predict dictionary. The wildcard character, an asterisk (*), may be used anywhere within the filename to create a list that matches the provided pattern. The asterisk (*) is a multi-character mask. Entering only an asterisk (*) selects all files.  |
| PREDICT<br>Dictionary | Is the one-character dictionary suffix for the Predict dictionary where the desired files are described. The available values are listed on the menu in parentheses and are determined by the ADABAS database administrator when AUTOADBS is installed. If only one dictionary is available (the suffix is blank), you will not be able to type a value in this field. |
| Master Target<br>PDS  | Is the fully qualified dataset name of the Master File PDS where the Master File is stored. Do not use quotation marks in the dataset name. Applies to MVS only.   |

## The AUTOADBS Facility

Access Target PDS	Is the fully qualified dataset name of the Access File PDS where the Access File is stored. Do not use quotation marks in the dataset name. Applies to MVS only.
FOCDEF Target PDS	Is the fully qualified dataset name of the FOCDEF file PDS where the TableTalk Help file is stored. Do not use quotation marks in the dataset name. Applies to MVS only, and this entry is only required if “Use COMMENTS for FOCDEF?” is specified as Yes (Y).
Replace Existing Description?	Indicates whether to overwrite existing Master and Access Files. No (N), the default, will not overwrite existing Master and Access Files. Type Yes (Y) to replace Master and Access Files that already exist on disk.
Include Fieldname Comments?	Indicates whether to exclude field comments in the Master File. Yes (Y), the default, includes field comments. Type No (N) to exclude them. Comments contain the full Predict dictionary fieldname, “*TRUNC*” if the name is truncated, the ADABAS format and length, and start/end characters for superdescriptor elements.
Use NATURAL Column Headings?	Indicates whether to use the NATURAL heading as the column heading in reports. No (N), the default, uses the fieldname for headings. Type Yes (Y) to use the NATURAL heading.
Use COMMENTS for FOCDEF?	Indicates whether to use Predict dictionary comments as help for TableTalk. No (N), the default, does not use Predict dictionary comments as help. Type Yes (Y) to use them. If you type Yes (Y) in MVS, a FOCDEF target PDS is required.
Start with Fieldname Position	Is a numeric value from 1 to 32. The default is 1. Type the starting position of the element name to be used as the first character in the fieldname. By increasing the starting position, you may strip off common prefixes from the element name.
for a Total Length of	Is a numeric value from 5 to 36. The default is 36. Type the maximum length of fieldnames described with AUTOADBS. Predict dictionary fieldnames are up to 32 characters and AUTOADBS can add 4-character suffixes to them.
Date Display Format	Is a valid date format for fields described in Predict dictionary as dates. The default is YYMD. Formats may include any valid combination of YY, Y, M, or D and can use month translation (T or TR) and day translation (W or WR). You may also display the day of week (W or WR), the quarter (Q, YQ, or YYQ), or Julian dates (JUL).

Display all Userviews?	Indicates whether to display all Master Files and views for the selected file(s). No (N), the default, does not display all Master Files and views. Type Yes (Y) to display them. This option is useful when selecting particular filenames. You may also toggle the display from the file list menu.
Parm File	Identifies the file containing your customized default values for the Main Menu.

The PF keys on the Main Menu perform the following functions:

**Help (PF1)**

Press the PF1 key to display an extensive online help facility.

**Refreshing the List of Files (PF2)**

Press the PF2 key to clear the existing list of files maintained by AUTOADBS for this session. When you select files from the Main Menu, information is gathered from the Predict dictionary. Each time you change the selection criteria, the new files are appended to the existing list. Pressing the PF2 key purges this list without exiting AUTOADBS. The message “Enter new file selection criteria” is displayed when the list is successfully purged. No message is displayed if you press the PF2 key before you have created a list of files.

**Exit (PF3)**

Press the PF3 key to exit AUTOADBS.

**Logging Default Menu Parameters (PF4)**

Press the PF4 key to save your customized default values for the Main Menu to disk. In MVS, these values are saved in member ADBS\$PRM in the parm dataset shown on the menu. This dataset is one of the following:

- The PDS pre-allocated to ddname ADBS\$PRM.
- The dataset *userid.FOCADBS.DATA*.
- The first dataset allocated to ddname FOCADBS.

Parameter logging assumes that you have write access to the target dataset. Attempting to log parameters to a dataset without write access will result in a security abend. See Section 4.5.8, *Search Order for Parameter Log File in MVS*, for further details.

In CMS, your customized default values are saved to ADBS\$PRM FOCADBS on your A disk when the PF4 key is pressed.

## The AUTOADBS Facility

The following information is logged:

- Filename
- Predict dictionary
- Master File partitioned dataset name (MVS only)
- Access File partitioned dataset name (MVS only)
- FOCDEF partitioned dataset name (MVS only)
- Replace existing description
- Include fieldname comments
- Use NATURAL column headings
- Create FOCDEF file
- Start with fieldname position
- For a total length of
- Date display format
- Display all views

Note that all validations must pass before the default values are logged.

### **TED Master File (PF5)/TED Access File (PF6)**

Press the PF5 or PF6 key to edit (using TED) the Master or Access File specified in the Master Filename entry field. MVS selects the member in the dataset named in either Master Target PDS or Access Target PDS. CMS selects the file “master MASTER A” or “master FOCADBS A.”

### **Picture of Master File (PF9)**

Press the PF9 key to generate a diagram of the structure of the file specified in the Master Filename entry field. After the picture is displayed, type in any character and press the Enter key to return to the Main Menu. In MVS, the master name must be a member of a dataset allocated to ddname MASTER to generate the picture (the Master Target PDS is not used).

### **File List (PF10)**

Press the PF10 key to display a list of all files that meet the screening criteria provided in the Master Filename entry field. The “Display all Userviews” prompt is not used for this report.

### 4.3.3 Generating the Master and Access Files

After you complete the Main Menu, press the Enter key to generate a Master and Access File. AUTOADBS informs you that it is accessing the dictionary with the following message:

```

**=====**
** Retrieving FILE information from dictionary **
** Please wait... **
**=====**

```

This message is only displayed during the first retrieval per AUTOADBS session or when the file selection criteria is changed after the previous retrieval (within the session). AUTOADBS does not access the dictionary a second time for the same immediate selections in a single session.

### The File Selection Screen

When file information retrieval is completed, the File Selection Screen displays.

```

Master: Master File Generation Facility for AUTOADBS Show userviews: Y
SAMPLE ==File Selection==

Select the ROOT file with 'R', children files with 'C'.
When a file exists in several databases, select the one entry that describes
the file/database combination desired.

R/C File A Fnr D
-----
r SYSDIC A 012
SYSDIC-DB U 012
SYSDIC-DDM U 012
SYSDIC-DEF U 012
SYSDIC-DESC U 012
SYSDIC-EL U 012
c SYSDIC-FI U 012
SYSDIC-FI-ADA U 012
SYSDIC-FI-GEN U 012
SYSDIC-IMS U 012
SYSDIC-KY U 012
SYSDIC-MO U 012

PF1=Help PF3=End PF4=Add Files PF5=Views PF7=Up PF8=Down

```

The File Selection Screen is used to choose the ADABAS files that will be described in the Master File. This screen displays all files that match the selection criteria provided on the Main Menu.

## The AUTOADBS Facility

If you select N (No) for “Display all Userviews” on the Main Menu, all files that exactly match the specifications supplied in the File Name field are displayed, whether they are ADABAS files or user views. For example, if you enter SYSDIC in the File Name field, the ADABAS file SYSDIC is displayed. If you enter SYSDIC-DB, only the view SYSDIC-DB is displayed. If you enter the wildcard character as part of a name (for example, SYSDIC\*), all files and views that begin with SYSDIC are displayed.

If you select Y (Yes) for “Display all Userviews” on the Main Menu, all files that exactly match the specifications supplied in the File Name field are displayed, along with all views associated with the selected files. For example, you will see the same results if you enter either SYSDIC or SYSDIC-DB in the File Name field because both files are views of the same ADABAS file.

To change your selection for “Display all Userviews” directly on the File Selection Screen, press the PF5 key. This will toggle the file list between only those files that exactly match your selections and all views associated with the selected files.

To add more ADABAS files to the list, press the PF4 key. When the Main Menu redisplay, select the next ADABAS filename you want to include and press the Enter key. You return to the File Selection Screen to see the additional file displayed in the list. Repeat this process for each additional file you want to add to the list.

Place an **r** in the column labeled “R/C” to select the root file. Select all other files to be used as descendants by placing a **c** in the “R/C” column. Press the Enter key when your file selection is complete.

The File Selection Screen displays the following information:

Master	Master Filename provided on the Main Menu.
Show userviews	Yes (Y) indicates that views are included on the list. No (N) indicates that views are not included.
R/C	Select the root file with R and children files with C.
File	Filename of selected files.
A	File type: A=ADABAS file, U=view.
Fnr	File number.
D	Predict dictionary suffix selected on Main Menu.

The PF keys on the File Selection Screen perform the following functions:

### **Help (PF1)**

Press the PF1 key to display an extensive online help facility.

### **Exit (PF3)**

Press the PF3 key to exit the File Selection Screen and return to the Main Menu.

**Add Files (PF4)**

Press the PF4 key to return to the Main Menu to add additional files to the list.

**Views (PF5)**

Press the PF5 key to toggle between including views and excluding them.

**Scroll Backward (PF7)/Scroll Forward (PF8)**

Press the PF7 key to scroll the contents of the File Selection Screen backward; press the PF8 key to scroll the contents forward.

After the File Selection Screen is completed, AUTOADBS informs you that it is accessing the dictionary for field information with the following message:

```
**=====**
**      Retrieving FIELD information from dictionary      **
**      Please wait...                                     **
**=====**
```

This message is displayed once for each Predict dictionary that must be accessed, based on the location of the selected files. This message does not appear if all selected files (or any of their associated views) have already been selected during this AUTOADBS session.



The Access File Attributes Screen displays the following information:

Master	Master Filename provided on the Main Menu.
File	Filename of selected file.
Dictionary	Predict dictionary suffix selected on Main Menu.
CALLTYPE	Type RL or FIND. The default is RL. If the site installation requires a CALLTYPE of RL, you cannot change this value.
PASSWORD	Type the password for password-protected files.
DBNR	Type the database number if you have access to multiple databases in your ADABAS environment. If this value is 0, the database number is determined from the DBID attribute in the ADABAS DDCARD available to your session.
S	Type S for the selected SEQFIELD.
Fieldname	Predict dictionary fieldname.
Alias	ALIAS used in the Master File. It is the ADABAS name.
Usage	USAGE format of the field in the Master File.
Actual	ACTUAL format of the field in the Master File.
D	Descriptor flag. Value is Descriptor (D) for descriptors.
U	Unique flag. Value is Unique (U) for unique descriptors.
S	Null suppression option. Values are Null suppression (N), Fixed format (F), or blank.
Ty	Field type. Values are Group (GR), Periodic group (PE), Multi-value field (MU), Superdescriptor (SP), or Subdescriptor (SB).
L	Level number. Values are from 1 to 7.

The PF keys on the Access File Attributes Screen perform the following functions:

**Help (PF1)**

Press the PF1 key to display an extensive online help facility.

**Restart (PF2)**

Press the PF2 key to cancel all previous selections and restart the Master File generation process at the File Selection Screen.

**Exit (PF3)**

Press the PF3 key to exit the Access File Attributes Screen and return to the Main Menu.

**Scroll Backward (PF7)/Scroll Forward (PF8)**

Press the PF7 key to scroll the contents of the Access File Attributes Screen backward; press the PF8 key to scroll the contents forward.

## The Child Selection Screen

If only one file is specified on the File Selection Screen, the Master and Access Files are generated and you return to the Main Menu.

Otherwise, you are asked to select the children of the root file on the Child Selection Screen. The Child Selection Screen lists files that you can select as descendants of the parent file.

```

Master:      Master File Generation Facility for AUTOADBS
SAMPLE      ==Child Selection==

File:   SYSDIC-DB                      Dictionary:

Select children files with 'S'
S  File                                Fnr Dictionary
-  -----
s  SYSDIC-FI                          012

PF1=Help  PF2=Restart  PF3=End  PF4=None  PF5=Picture      PF7=Up  PF8=Down
    
```

This screen is the first of three screens that display as part of the child identification process. For each child file selected, you provide a relationship between it and the parent file. You identify a common field in each file: the foreign key (IXFLD) in the child file and the primary key (KEYFLD) in the parent file. Choosing the keys is described in the sections, *The IXFLD Selection Screen* and *The KEYFLD Selection Screen*.

Place an **s** in the column labeled “S” next to each desired child file and press the Enter key.

Press the PF4 key if no files are to be assigned as children of the current parent segment.

The file relationships are described in top-down, left-to-right order. Initially, this screen displays the root file as the current parent. All files selected as children will appear in the list. Select only those files that will be children of the current parent by placing an **s** in the column labeled “S” for the desired files.

After identifying the primary and foreign keys for the root and all of its children, the process will repeat for the files selected as children of the root. The first child of the root will be displayed at the top of the screen. This is now the current parent file. All files selected as children on the File Selection Screen, but not yet assigned, appear in the list. Files from the list are selected as children of the current parent. These are grandchildren of the root.

Child selection continues down this first path until the user selects no descendants (by pressing the PF4 key). Only then will the second child of the previous parent be displayed at the top of the screen. This process continues until all files have been assigned children or all possible descendants have been exhausted. At this point, the description will be generated and the Main Menu will be redisplayed.

The Child Selection Screen displays the following information:

Master	Master Filename provided on the Main Menu.
File	The current file being assigned children.
Dictionary	The Predict dictionary where this file is described.
S	Type S to select the children files.
File	Files to be assigned as children for the shown parent.
Fnr	File number.
Dictionary	Predict dictionary where this file is described.

The PF keys on the Child Selection Screen perform the following functions:

**Help (PF1)**

Press the PF1 key to display an extensive online help facility.

**Restart (PF2)**

Press the PF2 key to cancel all previous selections and restart the Master File generation process at the File Selection Screen.

**Exit (PF3)**

Press the PF3 key to exit the Child Selection Screen and return to the Main Menu.

**None (PF4)**

Press the PF4 key if you do not want to describe any of the listed files as children of the current parent file.

**Create a Picture of the Description (PF5)**

Press the PF5 key at any time to generate a diagram of the structure of the file being described. Note that the description is created on disk (and remains there even if the program is ended by pressing the PF3 key). After the picture is displayed, type in any character and press the Enter key to return to the Child Selection Screen. In MVS, the target Master PDS must be allocated to ddname MASTER to generate the picture. If a member exists with the master name selected in a dataset concatenated in front of the target dataset, the picture is generated from that member.

**Scroll Backward (PF7)/Scroll Forward (PF8)**

Press the PF7 key to scroll the contents of the Child Selection Screen backward; press the PF8 key to scroll the contents forward.

## The IXFLD Selection Screen

When files are selected as children, the IXFLD Selection Screen displays, once for each selected file.

```

Master:      Master File Generation Facility for AUTOADBS
SAMPLE                                ==IXFLD Selection==

File:  SYSDIC-FI                      Dictionary:
Parent: SYSDIC-DB                     Dictionary:

Select the common field (IXFLD) from the descendant file with 'S'
S  Fieldname                          Alias      Usage  Actual D U S Ty L
-  - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
s  HOLD_USER                           AF         A8     A8     D   N   1
   FILE_REC                             S6         A33    A33    D   N SP 1
   L_FNR                                AR         P3     Z3     D   N   1
   ADANET_FILE_TYPE                     D8         A1     A1     D   N   2
   ADABAS_AVB_REC                        S4         A34    A34    D   N SP 1
   IMS_SEGMENT_NAME                      AX         A8     A8     D   N   2
   IMS_PARENT_FILE                       AY         A32    A32    D   N   2
   IMS_SOURCE_FILE1                      A2         A32    A32    D   N   2
   IMS_SOURCE_FILE2                      A3         A32    A32    D   N   2
   DB2_EDIT_PROGRAM                      D9         A8     A8     D   N   2
   DB2_VALID_PROGRAM                     EA         A8     A8     D   N   2
   DSNAME                                 DL         A44    A44    D   N   2

PF1=Help  PF2=Restart  PF3=End  PF4=Skip                                PF7=Up  PF8=Down
    
```

The IXFLD Selection Screen identifies the foreign key in the child file. Place an **s** in the column labeled “S” to select the common field in the cross-reference file and press the Enter key.

Press the PF4 key to skip this file as a descendant if it was selected in error. Pressing the PF4 key does not affect previous or subsequent selections.

AUTOADBS displays only those fields that can be used as an IXFLD on this screen. The IXFLD must be a descriptor, a superdescriptor, or a subdescriptor. It cannot be a PE group, an element of a PE group, an MU field, or a reformatted field.

The IXFLD Selection Screen displays the following information:

- Master            Master Filename provided on the Main Menu.
- File             The child file from which the IXFLD is being selected.
- Dictionary       Predict dictionary where this file is described.
- Parent           The parent filename.
- Dictionary       Predict dictionary where this file is described.
- S                Type S for the selected IXFLD.

Fieldname	Predict dictionary fieldname.
Alias	ALIAS used in the Master File. It is the ADABAS name.
Usage	USAGE format of the field in the Master File.
Actual	ACTUAL format of the field in the Master File.
D	Descriptor flag. Value is Descriptor (D) for descriptors.
U	Unique flag. Value is Unique (U) for unique descriptors.
S	Null suppression option. Values are Null suppression (N), Fixed format (F), or blank.
Ty	Field type. Values are Group (GR), Periodic group (PE), Multi-value field (MU), Superdescriptor (SP), or Subdescriptor (SB).
L	Level number. Values are from 1 to 7.

The PF keys on the IXFLD Selection Screen perform the following functions:

**Help (PF1)**

Press the PF1 key to display an extensive online help facility.

**Restart (PF2)**

Press the PF2 key to cancel all previous selections and restart the Master File generation process at the File Selection Screen.

**Exit (PF3)**

Press the PF3 key to exit the IXFLD Selection Screen and return to the Main Menu.

**Skip (PF4)**

Press the PF4 key to skip this file as a descendant if it was selected in error. Previous or subsequent selections are not affected; however, the “skipped” segments appear in subsequent lists when you return to the Child Selection Screen to define any additional paths in the hierarchy.

**Scroll Backward (PF7)/Scroll Forward (PF8)**

Press the PF7 key to scroll the contents of the IXFLD Selection Screen backward; press the PF8 key to scroll the contents forward.

## The KEYFLD Selection Screen

Once the IXFLD is selected in the child file, the KEYFLD Selection Screen displays, once for each selected file.

```

Master:      Master File Generation Facility for AUTOADBS
SAMPLE      ==KEYFLD Selection==

File:       SYSDIC-DB           Dictionary:
Child:      SYSDIC-FI         Dictionary:
IXFLD:     FILE_REC          Actual: A33
Select the common field (KEYFLD) from the parent file with 'S'
 S  Fieldname                    Alias      Usage  Actual  D U S Ty L
  -  - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      RECORD_TYPE                AA         A2     A2           N  1
      COMMENTS                   BD         A30    A30          N MU 1
      DESC                       AB         A1     A1           N  1
      MAINTENANCE_ACTION         BG         A1     A1           N  2
      KEY_WORD                   KY         A32    A32          D N MU 1
      OWNER_ID                   II         A32    A32          D N MU 1
      DATA_BASE_NAME            JJ         A32    A32           N  1
      SPMRF_TYPE                 DA         A1     A1           N  1
  s  FILE_ELEMENT                DC         A32    A32           N MU 1
      FILE_TYPE_EL               S7         A33    A33          D N SP 1
      DATA_BASE_TYPE            AT         A1     A1           D N  1
      GEN_FLAG                   CI         A1     A1           N MU 2

PF1=Help  PF2=Restart  PF3=End  PF4=Skip                PF7=Up  PF8=Down
    
```

The KEYFLD Selection Screen identifies the primary key in the parent file. Place an **s** in the column labeled “S” to select the common field in the parent file and press the Enter key.

Press the PF4 key to skip this parent/child relationship if it was selected in error. Pressing the PF4 key does not affect previous or subsequent selections.

AUTOADBS displays only those fields that can be used as a KEYFLD on this screen. The KEYFLD must have the same format type as the IXFLD, and the field length may be the same or shorter than that of the IXFLD. It cannot be a periodic group (PE), a subdescriptor, a superdescriptor composed of partial fields, or a reformatted field.

The KEYFLD Selection Screen displays the following information:

- Master            Master Filename provided on the Main Menu.
- File            The parent file from which the KEYFLD is being selected.
- Dictionary      Predict dictionary where the parent file is described.
- Child           The child file from which the IXFLD was selected.
- Dictionary      Predict dictionary where this child file is described.

IXFLD	The selected IXFLD.
Actual	The ACTUAL format of the selected IXFLD.
S	Type S for the selected KEYFLD.
Fieldname	Predict dictionary fieldname.
Alias	ALIAS used in the Master File. It is the ADABAS name.
Usage	USAGE format of the field in the Master File.
Actual	ACTUAL format of the field in the Master File.
D	Descriptor flag. Value is Descriptor (D) for descriptors.
U	Unique flag. Value is Unique (U) for unique descriptors.
S	Null suppression option. Value are Null suppression (N), Fixed format (F), or blank.
TY	Field type. Values are Group (GR), Periodic group (PE), Multi-value field (MU), Superdescriptor (SP), or Subdescriptor (SB).
L	Level number. Values are from 1 to 7.

The PF keys on the KEYFLD Selection Screen perform the following functions:

**Help (PF1)**

Press the PF1 key to display an extensive online help facility.

**Restart (PF2)**

Press the PF2 key to cancel all previous selections and restart the Master File generation process at the File Selection Screen.

**Exit (PF3)**

Press the PF3 key to exit the KEYFLD Selection Screen and return to the Main Menu.

**Skip (PF4)**

Press the PF4 key to skip this file as a descendant if it was selected in error. Previous or subsequent selections are not affected; however, the “skipped” segments appear in subsequent lists when you return to the Child Selection Screen to define any additional paths in the hierarchy.

**Scroll Backward (PF7)/Scroll Forward (PF8)**

Press the PF7 key to scroll the contents of the KEYFLD Selection Screen backward; press the PF8 key to scroll the contents forward.

## 4.4 Background Execution

---

You can execute AUTOADBS in the background as a batch job to create a Master File from a single ADABAS record by passing values normally provided on the menus as arguments. The syntax is

```
EX AUTOADBS BATCH=Y,USER=userid,MASTER=master,FILE_NAME=filename,
      SEQFIELD=seqfield [,options]
```

where:

<i>userid</i>	Is the high-level qualifier for output datasets. It is required if target dataset names are not provided.
<i>master</i>	Is the 1- to 8-character name for the file description.
<i>filename</i>	Is the 1- to 32-character ADABAS filename.
<i>seqfield</i>	Is the fieldname to be used as the SEQFIELD.
<i>options</i>	Are provided on the command line by entering the name-value pair and separating them with commas. Options may extend over several lines. User-provided values and default values are echoed after the Master File is generated. Choose one or more of the following:
CALLTYPE={ <u>RL</u>  FIND}	Indicates whether the Interface constructs Read Logical (RL) or FIND calls for retrieval. Default is RL.
PASSWORD= <i>password</i>	Is the file password. If the password is not provided, it will not be included in the Access File.
DBID= <i>nnn</i>	Is the database number. If the DBID is not provided, it will not be included in the Access File.
DICT= <i>suffix</i>	Uses the custom Predict dictionary. It is a site installation option.
REPLACE={Y N}	Replaces existing Master and Access Files. Default is No (N).
COMMENTS={ <u>Y</u>  N}	Includes fieldname comments. Default is Yes (Y).
HEADINGS={Y N}	Uses NATURAL column headings. Default is No (N).
FOCDEF={Y N}	Creates FOCDEF file. Default is No (N).
START= <i>nn</i>	Is the fieldname start position. Default is 1.

LENGTH= <i>nn</i>	Is the maximum field length. Default is 36.
DATEDISP= <i>format</i>	Is the date display format. Default is YYMD.
REQ_SEQFIELD=NO	Is used only when SEQFIELD is not known.
MASTERDATA= <i>dsn</i>	Is the Master target PDS. Default is <i>userid.MASTER.DATA</i> .
FOCADBSDATA= <i>dsn</i>	Is the Access target PDS. Default is <i>userid.FOCADBS.DATA</i> .
FOCDEFDATA= <i>dsn</i>	Is the FOCDEF target PDS. Default is <i>userid.FOCDEF.DATA</i> .

**Note:**

- SEQFIELD will be ignored if the named field cannot be found in the dictionary for the selected file or the field cannot be used as a SEQFIELD, and the installation does not require a SEQFIELD.
- CALLTYPE will be set to RL by default. If the site installation requires a CALLTYPE of RL, you cannot change this value. See Appendix A, *Installation Instructions*, for more information.
- PASSWORD will be ignored if its length exceeds eight characters.
- DBID is used to populate the DBNO attribute in the Access File.

## 4.5 The Generated Descriptions

---

AUTOADBS creates complete Master and Access Files for your selected files. The following sections describe the attributes and values assigned to them by AUTOADBS.

### 4.5.1 Master File and Segment Attributes

---

File and segment declarations include:

FILENAME=	Is the Master Filename specified on the Main Menu.
SUFFIX=	Is always ADBSINX.
SEGNAME=	Indicates the segment names in the Master File generated by AUTOADBS that follow a logical format to provide uniqueness within the file. For the root segment of the ADABAS file (ACCESS=ADBS), the segment name is <i>Snn</i> , where <i>nn</i> is a two-digit number that indicates the order in which the file was selected. The first file (selected as the root from the file selection menu) has SEGNAME=S01. Subsequent files used in the same description (selected as children from the file selection menu) will have SEGNAME=S02, SEGNAME=S03, and so on.  The segment names for PE groups and MU fields have the format <i>aammnn</i> , where <i>aa</i> is the ADABAS name of the field, <i>mm</i> is a two-digit number that indicates the order in which the PE group or MU field appears in the Predict dictionary description of the file, and <i>nn</i> is the order number used for the root segment. For example, SEGNAME=BE0201 describes the segment for the field BE, which is the second (02) PE group or MU field described in the first segment (01).
SEGTYPE=	Is the root segment and cross-reference segment type. The SEGTYPE is S for the root file and children with a non-unique IXFLD. The SEGTYPE is U for a child with a unique IXFLD. The SEGTYPE for all PE and MU fields is S.
PARENT=	Is the value of SEGNAME of the parent record.
OCCURS=	Indicates the number of occurrences of a PE group or MU field. This attribute will contain the ADABAS fieldname of the PE group or MU field with the suffix C, which is the ALIAS of the counter field in the parent segment.
MAX=	Indicates the maximum number of occurrences of the PE group or MU field as described in the Predict dictionary. This attribute is a comment only and is not used by the ADABAS Interface.

## 4.5.2 Access File Attributes

---

Access File declarations include:

RELEASE=	Indicates the ADABAS release number (for documentation only).
OPEN=	Determines if the ADABAS Interface issues ADABAS OPEN and CLOSE calls for each report request. Yes (Y) is the default.
SEGNAM=	Is the segment name as described in the Master File.
ACCESS=	Specifies the access method for the segment. ADBS indicates segments that contain non-repeating data. PE indicates segments that describe periodic groups. MU indicates segments that describe multi-value fields.
DBNO=	Is the ADABAS database number. This attribute will not be included in the Access File if a value was not provided during the AUTOADBS execution.
FILENO=	Is the ADABAS file number.
PASS=	Is the ADABAS password for the file. This attribute is only included if a value was provided during the AUTOADBS execution.
CALLTYPE=	Determines whether FIND or RL calls are made to retrieve the data.
SEQFIELD=	Is the SEQFIELD that you selected. This attribute will not be included if a SEQFIELD was not selected.
KEYFLD=	Is the common field in the parent segment used to relate two files.
IXFLD=	Is the common field in the child segment used to relate two files.
FIELD=	Describes a descriptor, superdescriptor, subdescriptor, or a component of a superdescriptor.
TYPE=	Indicates the type of descriptor for FIELD. Values are superdescriptors (SPR), subdescriptors or superdescriptors composed of partial fields (NOP), or a simple descriptor (DSC).
NU=	Indicates if the field uses null suppression. Values are Yes (Y) or No (N).

## 4.5.3 Master File Field Attributes

---

Field declarations include:

FIELD=	Is the fieldname from the Predict dictionary.
ALIAS=	Is the actual ADABAS name.
USAGE=	Is the FOCUS USAGE format of the field. This attribute determines how the value is displayed in reports.
ACTUAL=	Is the field format as stored in the ADABAS file.

## The AUTOADBS Facility

INDEX=I	Indicates that the field is a superdescriptor, subdescriptor, or simple descriptor.
TITLE=	Is the column heading used in reports. This attribute is included only if the field has a column heading value in the Predict dictionary and the menu option “Include NATURAL Column Headings” was selected.
GROUP=	Identifies fields described as simple groups or PE groups.
\$GROUP=	Is a group field that is itself part of another group. Imbedded groups are not supported by the ADABAS Interface.
\$GRMU=	Is a group that contains a PE group or an MU field.
\$2LONG=	Is a group field whose total length exceeds the maximum of 256 characters supported by FOCUS.
\$PEMU=	Is a PE group that contains an MU field or is a group field that contains a PE group or an MU field. In both cases, the field cannot be used to retrieve data using the ADABAS Interface.
\$PEDIF=	Is a PE group described in a view in which all of the component elements that are described in the Master File are not included in the view. As a result, the length of the PE group cannot be determined accurately.
\$NOMAS=	Is a field described in a Predict dictionary view that does not exist in the Master File. Typically, this group field combines several real fields from the Master File in the view. The ADABAS name associated with this field does not exist in the physical ADABAS file’s FDT.
\$REDEF=	Is a redefinition of an actual ADABAS field. The field itself does not exist in the ADABAS file and does not have an ADABAS name in the Predict dictionary.
\$PH=	Is a phonetic descriptor. Phonetic descriptors are not supported by the ADABAS Interface.
\$HY=	Is a hyper descriptor. Hyper descriptors are not supported by the ADABAS Interface.
\$HM=	Is a hyper descriptor used as an MU field. Hyper descriptors are not supported by the ADABAS Interface.
\$HP=	Is a hyper descriptor used as a PE group. Hyper descriptors are not supported by the ADABAS Interface.
<i>\$fieldname</i>	Is found at the end of the Master File and identifies duplicate fields. It lists the fieldname as found in the Master File, the number of duplicate occurrences, the segments in which the duplicate fields are located, and the full Predict dictionary fieldname.

## 4.5.4 Fieldnames

---

Fieldnames are derived from the Predict dictionary and may be truncated based on your selections on the Main Menu. Hyphens in the Predict dictionary fieldname are converted to underscores. AUTOADBS also creates several fields in the generated description, both for ease of use and Interface requirements. These fields include counter fields, order fields, superdescriptor elements, DEFINEd fields, and cross-reference fields.

- A counter field is generated for each PE group and MU field in the ADABAS file. This field is used to determine the number of occurrences of the PE group or MU field.

A counter field is located in the repeating group's parent segment and is referenced in the OCCURS attribute of the repeating group's segment. This process is accomplished by appending the letter C to the ADABAS name in the ALIAS attribute of the file description.

You refer to this field in report requests when you want to print the number of occurrences of the repeating group.

The FIELDNAME of the counter field has the format *aammnn\_CNT*, where *aammnn* is the same as the name of the segment in which the field is described.

- An order field is generated for each repeating group and is added at the end of the repeating group's segment.

This field has an ALIAS of ORDER and is used to determine the sequence of a particular repeating field.

The FIELDNAME of the order field has the format *aammnn\_OCC*, where *aammnn* is the same as the name of the segment in which the field is described.

- For superdescriptors, AUTOADBS generates unique fieldnames for the element fields. A suffix in the format *\_Snn* is appended to the element fieldname, where *nn* is incremented by 1 for each superdescriptor in the ADABAS file.

This suffix allows AUTOADBS to list the element fields with minimal risk of creating duplicate fieldnames, particularly in cases where the element fields are used in several superdescriptors.

Note that the element fields are also described in the Master File. If the element field appears in only one superdescriptor, you may edit the Master File to remove the original occurrence of the field and rename the superdescriptor entry.

Perform any selection tests on the element field (with the *\_Snn* suffix) rather than the original field to take advantage of implicit use of the superdescriptor inverted list.

## The AUTOADBS Facility

- AUTOADBS generates a Master File-based DEFINE for NATURAL fields described as dates. FOCUS converts the internal ADABAS date value to a FOCUS date. This DEFINED fieldname has the format *fieldname\_DAT*, where *fieldname* is the name from the Predict dictionary.

**Note:** To better understand the allowable FOCUS date formats, refer to the *FOCUS for IBM Mainframe User's Manual*.

- AUTOADBS also generates a field with the format *aa\_NOP*, where *aa* is the ADABAS name of the field. This field is generated only when an embedded group field is selected as a SEQFIELD, KEYFLD, or IXFLD. This field is also included in the Access File for the appropriate attribute. Normally, you do not refer to this field in your report requests.

## Aliases

---

The ALIAS generated by AUTOADBS is typically the ADABAS fieldname. In certain situations, however, the ALIAS is altered. These situations include:

- Fields generated as counters.
- Superdescriptor elements that are PE groups or MU fields.
- Fields with ADABAS formats not directly supported in FOCUS.
- Fields that have different formats in several views.

In the simplest case, counter fields for PE groups and MU fields are generated where the alias is the ADABAS name with the letter C appended.

For superdescriptor elements that are PE groups or MU fields, the alias consists of the ADABAS name appended with the digit 1. This ADABAS syntax indicates that the first field of the repeating group will be retrieved, rather than all values in the group. Normally, you do not reference these fields in a report request. Instead, use the original field. This syntax is provided only to allow record selection against a superdescriptor that has both alphanumeric and numeric elements.

The syntax for selecting against these fields is

```
WHERE super EQ 'aaa/nnnF/aaa'
```

where:

<i>super</i>	Represents the superdescriptor that has both alphanumeric and numeric elements.
<i>aaa</i>	Represents the alphanumeric portion of the group.
<i>nnnF</i>	Represents the numeric portion of the group. The sign value of F is required.

Fields with ADABAS formats not directly supported in FOCUS and fields that have different formats in several views are reformatted by ADABAS before being presented to FOCUS. This reformatting is accomplished by adding ADABAS formatting options to the alias in the format *aa,l,t*, where *aa* is the ADABAS name of the field, *l* is the length to be used, and *t* is the ADABAS format.

## 4.5.5 USAGE and ACTUAL Formats

The following table illustrates how the ADABAS format and length are converted to the FOCUS USAGE and ACTUAL formats by AUTOADBS. In cases where the ADABAS field must be reformatted to a supported FOCUS format, the ALIAS is provided to show how this reformatting is accomplished.

Predict Format	Predict Length	FOCUS USAGE	FOCUS ACTUAL	FOCUS ALIAS	Comments
A	n	An	An	aa	
B	n	An	An	aa	(Super and subdescriptors)
B	1	I4	I1	aa	
B	2	I6	I2	aa	
B	3	I8	I4	aa,4,B	
B	4	I9	I4	aa	
B	5	P12	P7	aa,7,P	
B	6	P15	P8	aa,8,P	
B	>6	An	An	aa,n,A	where n=(length * 2) + 2
D		P7	P4	aa	defined date field created
L		I1	I1	aa	
T		P13	P7	aa	
F	4	F9.2E	F4	aa	
F	8	D15.2E	D8	aa	
I	1	I4	I1	aa	
I	2	I6	I2	aa	
I	4	I9	I4	aa	
I	8	A18	A18	aa,20,A	
N	n	Pn	Zn	aa	
U	n	Pn	Zn	aa	
P	n	Pn	Pk	aa	k=(n+2)/2
N	n.m	Pj.m	Zk	aa	j=n+m+1, k=n+m
U	n.m	Pj.m	Zk	aa	j=n+m+1, k=n+m
P	n.m	Pj.m	Pk	aa	j=n+m+1, k=(n+m+2)/2
NS	n	Pj	Zk	aa	j=n+m+1, k=n+m
US	n	Pj	Zk	aa	j=n+m+1, k=n+m
PS	n	Pj	Pk	aa	j=n+m+1, k=(n+m+2)/2
NS	n.m	Pj.m	Zk	aa	j=n+m+2, k=n+m
US	n.m	Pj.m	Zk	aa	j=n+m+2, k=n+m
PS	n.m	Pj.m	Pk	aa	j=n+m+2, k=(n+m+2)/2

## 4.5.6 Changes to the Generated Descriptions

---

In general, you do not need to change the descriptions generated by AUTOADBS. However, you do need to edit the generated Master File and/or its corresponding Access File if:

- Duplicate fieldnames exist.
- The DBNO or FILENO included in the Predict dictionary is different from the actual ADABAS file.
- You use different fieldnames from those in the Predict dictionary.
- You add edit display options to fields.
- You increase the display length of numeric fields.
- You change OPEN=YES to OPEN=NO in the Access File.
- You replace AUTOADBS generated superdescriptor elements that occur only once in the file with the original field.
- You add Master File-based DEFINES to describe redefined fields included in the Predict dictionary.
- You remove segments because the 64-segment FOCUS limitation has been exceeded. This step is necessary when AUTOADBS generates the following completion message: DESCRIPTION GENERATED - nnn SEGMENTS DESCRIBED. Typically, you remove PE groups or MU field segments that are not required. Alternatively, you may remove the segment and add the repeating field to the non-repeating segment, once for each anticipated occurrence of the field.

For example, if you anticipate three occurrences, you add three fields to the Master File (with unique fieldnames). The ALIAS has the format *aan*, where *aa* is the ADABAS name of the field and *n* is the relative entry number from 1 to 3. MU fields that are part of a PE group are described in the non-repeating segment with an ALIAS in the format *aan(m)*, where *aa* is the ADABAS name of the field, *n* is the relative entry number of the PE group, and *m* is the relative entry number of the MU field within the PE group.

## 4.5.7 Creating a Record of Generated Master and Access Files

---

AUTOADBS maintains a temporary list of the Master and Access Files generated during any one session. This list is refreshed at the beginning of each session and is normally erased at the end of the session. You retain this list by executing AUTOADBS with the following syntax:

```
EX AUTOADBS MFDLIST=Y
```

In MVS, the list resides in a temporary dataset allocated to ddname AUTOADBL, with a disposition of MOD and record length of 114.

In CMS, the list resides in the file AUTOADBL FOCTEMP with a record length of 26.

It is your responsibility to free or erase this temporary file when it is no longer required. The file layout is:

<b>Length</b>	<b>Columns</b>	<b>Description</b>
8	1-8	Master Filename
8	9-16	Number of duplicate fieldnames
5	17-21	Number of segments described
5	22-26	Blank
44	27-70	Master target PDS name (MVS only)
44	71-114	Access target PDS name (MVS only)

## 4.5.8 Search Order for Parameter Log File in MVS

---

The AUTOADBS parameter log file is stored in a PDS with member name ADBS\$PRM. If the member does not exist, it is created for you. For MVS, the PDS is located using one of three methods, in the following order:

1. DDNAME ADBS\$PRM

If this ddname is allocated to a PDS prior to execution of AUTOADBS, it is used. The member ADBS\$PRM is used or created for you.

If this ddname is allocated to a sequential dataset, it is freed, a message is generated, and parameter logging is disabled.

If DDNAME ADBS\$PRM is not allocated, AUTOADBS continues searching and attempts to allocate it in the steps that follow. AUTOADBS does not free this ddname upon exiting, assuming that it may be used again, even if it was allocated by AUTOADBS with one of the following two methods.

2. Dataset name *userid*.FOCADBS.DATA

This name is the default provided in the AUTOADBS FOCEXEC as &DSNP0. If that default was changed during the ADABAS Interface installation, the dataset name you supply is used.

This dataset must be a PDS. If it is not, a message is displayed and parameter logging is disabled.

3. DDNAME FOCADBS

The first dataset allocated to ddname FOCADBS is used as the parameter file if the first two methods fail.

If ddname FOCADBS is not allocated, a message is displayed and AUTOADBS exits. AUTOADBS cannot be executed if there is no FOCADBS that describes the Predict dictionary files.

Method number two assumes standard IBI naming conventions. Method number three assumes that your dataset is allocated first in the concatenation of datasets to ddname FOCADBS. The first method allows you to identify the profile dataset prior to execution of AUTOADBS. This option is recommended for sites that have nonstandard dataset naming conventions.

## 5 Describing ADABAS Files to FOCUS

---

### Topics:

- Master Files
- Access Files

Before you can access an ADABAS file using FOCUS, you must first describe it to FOCUS in two files: a Master File, and an associated Access File. A Master File is required to describe the ADABAS file in standard FOCUS terminology to the ADABAS Interface. The Access File describes the database access information, for example, database number, file number, and descriptor fields.

Under MVS, the Master File is a member of a PDS with ddname MASTER. The Access File is a member of a PDS with ddname FOCADBS or ACCESS. Under CMS, the Master File is stored with a filetype of MASTER. The Access File is stored with a filetype of FOCADBS or ACCESS.

Analyze your ADABAS structures to decide how you want to describe them to FOCUS. Once that is done, your next step is to create their Master and Access Files. The preferred way to create Master and Access Files is to use the AUTOADBS facility described in Chapter 4, *The AUTOADBS Facility*. This chapter is helpful if your site does not have the Predict dictionary or if you want to modify the Master and Access Files.

FOCUS processes a report request with the following steps:

1. It locates the Master File for the ADABAS file. The filename in the report request is the same as the MVS PDS member name or CMS filename for filetype MASTER.
2. It detects the SUFFIX attribute, ADBSINX, in the Master File. Since this value indicates that the data resides in an ADABAS file, it passes control to the Interface.
3. The Interface locates the corresponding Access File and uses the information contained in both the Master and Access Files to generate the ADABAS direct calls required by the report request. It passes these direct calls to the ADABAS DBMS.
4. The Interface retrieves the data generated by the ADABAS DBMS and returns control to FOCUS. For some operations, FOCUS may perform additional processing on the returned data.

## 5.1 Master Files

FOCUS permits a great deal of flexibility in its Master Files. For the ADABAS Interface, you need only describe the ADABAS fields actually used in reporting applications, omitting any unnecessary ADABAS fields. (Note the exception that if a PE group is included in the Master File, all fields of the PE group must be included. See Section 5.1.3, *Field Attributes*, and Section 5.1.7, *Describing Multi-value Fields Within Periodic Groups*, for additional information.) Additionally, describing the same ADABAS file in multiple ways enables you to access that same file in its different configurations within one FOCUS procedure.

FOCUS Master Files have three parts: file declaration, segment declaration, and field declaration, each of which is explained in this chapter. Concepts which are specific to ADABAS files, for which FOCUS requires unique syntax in the Master File, are also explained.

The following is an example of the Master File for VEHICLES.

```

$$$ CREATED BY AUTOADBS ON 12/10/97 AT 10.17.27 BY PMSMJB
FILENAME=ADACAR , SUFFIX=ADBSINX , $

$ ADABAS FILE = VEHICLES-FILE                                DICTIONARY = 6
SEGNAME=S01          , SEGTYPE=S , $
FIELD= REG_NUM              , ALIAS= AA          , A15  , A15  , INDEX=I , $
FIELD= CHASSIS_NUM         , ALIAS= AB          , I9    , I4    , $
FIELD= PERSONNEL_ID        , ALIAS= AC          , A8    , A8    , INDEX=I , $
GROUP= CAR_DETAILS        , ALIAS= CD          , A50   , A50   , $
  FIELD= MAKE              , ALIAS= AD          , A20   , A20   , INDEX=I , $
  FIELD= MODEL             , ALIAS= AE          , A20   , A20   , $
  FIELD= COLOR             , ALIAS= AF          , A10   , A10   , INDEX=I , $
FIELD= YEAR                , ALIAS= AG          , P2    , Z2    , $
FIELD= CLASS               , ALIAS= AH          , A1    , A1    , INDEX=I , $
FIELD= LEASE_PUR           , ALIAS= AI          , A1    , A1    , $
FIELD= DATE_ACQ           , ALIAS= AJ          , P6    , Z6    , $
$GRMU = CAR_MAINTENANCE  , ALIAS= AK          , A11   , A7    , $
  FIELD= CURR_CODE         , ALIAS= AL          , A3    , A3    , $
  FIELD= MAINT_COST_CNT    , ALIAS= AMC         , I4    , I2    , $
FIELD= DAT_ACQ_DESC       , ALIAS= AN          , A4    , A4    , INDEX=I , $
GROUP= MODEL_YEAR_MAKE   , ALIAS= AO          , A28   , A22   , INDEX=I , $
  FIELD= YEAR_S02         , ALIAS= AG          , P2    , Z2    , $
  FIELD= MAKE_S02         , ALIAS= AD          , A20   , A20   , INDEX=I , $

SEGNAME=AM0101  , SEGTYPE=S , PARENT=S01  , OCCURS=AMC , $  MAX= 60
FIELD= MAINT_COST          , ALIAS= AM          , P7    , P4    , $
FIELD= AM0101_OCC         , ALIAS= ORDER      , I4    , I2    , $

```

**Figure 5-1. Example of the VEHICLES Master File**

## 5.1.1 File Attributes

---

Master Files begin with a file declaration that names the file and describes the type of data source. The file declaration has two attributes, FILENAME and SUFFIX.

The syntax is

```
FILE[NAME]= filename, SUFFIX=ADBSINX [, $]
```

where:

*filename* Is a one- to eight-character name that complies with FOCUS file naming conventions.

ADBSINX Is the value that specifies the ADABAS Interface.

### FILENAME

---

The FILENAME (or FILE) is any valid name, as defined in the *FOCUS for IBM Mainframe User's Manual*, from one to eight characters long.

In the MVS environment, it is a member of a partitioned dataset with the ddname MASTER; in the CMS environment, it is stored with the filetype MASTER.

### SUFFIX

---

The value for the SUFFIX attribute is always ADBSINX, which is the name of the ADABAS Interface program that FOCUS loads to read an ADABAS database.

### Example

---

The following example is a typical file declaration:

```
FILENAME=ADACAR, SUFFIX=ADBSINX, $
```

## 5.1.2 Segment Attributes

---

Each segment declaration describes one ADABAS record or a subset of an ADABAS record (called a logical record type). Each logical record type described in a Master File requires a segment declaration that consists of at least two attributes, SEGNAME and SEGTYPE.

The syntax is

```
SEGNAME= segname, SEGTYPE=segtype [, PARENT=parent][, OCCURS=occursname][, $]
```

where:

<i>segname</i>	Is a unique one- to eight-character name that identifies the segment.
<i>segtype</i>	Identifies the characteristics of the segment. Possible values are:  S            Indicates multiple instances of a descendant segment are related to a given parent.  U            Indicates a single instance of data for a descendant segment is related to its parent.  KL           Indicates multiple instances of a cross-referenced descendant segment are related to a given parent.  KLU          Indicates a single instance of a cross-referenced descendant segment is related to its parent.
<i>parent</i>	Is the name of the parent segment from the ADABAS database.
<i>occursname</i>	Indicates a data field in the parent segment that contains the number of occurrences of the descendant segment. Its value is derived from the two-character internal ADABAS name (FOCUS ALIAS field) appended with the letter C.

### SEGNAME

---

The SEGNAME (or SEGMENT) attribute is the name of a group of data fields that have a one-to-one relationship to each other in the ADABAS file.

You may describe an ADABAS file in the FOCUS hierarchical design to take advantage of multi-value fields and periodic groups. You may have multiple ADABAS records described in the same FOCUS view. Each record is described as a segment with a unique SEGNAME.

See your FOCUS for IBM mainframe documentation for more information on the number of segments allowed in a Master File.

## SEGTYPE

---

The SEGTYPE attribute identifies the basic characteristics of related or cross-referenced segments. Cross-referencing is a method of accessing information from two or more different files or segments to use in a single report request. Refer to the *FOCUS for IBM Mainframe User's Manual* for more information on file and segment cross-referencing.

## PARENT

---

Any segment except the root is a descendant, or child, of another segment. The PARENT attribute supplies the name of the segment which is the logical parent or owner of the current segment. If no PARENT attribute appears, the default is the immediately preceding segment; however, it is highly recommended to include the parent. The PARENT name is the one- to eight-character SEGNAME of a previous segment.

## OCCURS

---

For more information on the segment attribute OCCURS, which applies to repeating fields and groups, see Section 5.1.6, *Describing Multi-value Fields and Periodic Groups With the OCCURS Attribute*.

## Example

---

The following two examples are typical segment declarations:

```

SEGNAME=S01          ,SEGTYPE=S          , $
SEGNAME=AM0101      ,SEGTYPE=S          ,PARENT=S01      ,OCCURS=AMC      , $

```

### 5.1.3 Field Attributes

---

Field declarations describe the fields in each file. For the simplest file structures (single-segment, fixed-length files), you only need to describe the ADABAS fields you actually use for reporting purposes.

Put them in any order within their segment, except in the case of fields within periodic groups (PE). The PE group must have all fields defined and in the proper order.

The syntax is

```
FIELD[NAME]=fieldname, ALIAS=alias , [USAGE=] display, [ACTUAL=] format, $
```

where:

*fieldname* Is the 1- to 66-character name that is unique to the Master File.

*alias* Is the two-character internal ADABAS fieldname.

*display* Is the FOCUS display format for the field.

*format* Is the FOCUS definition of the ADABAS field format and length.

**Note:** Field declarations are always terminated with , \$.

There are additional attributes that apply to superdescriptor, cross-referenced, repeating, and other types of fields. These attributes are discussed later in this chapter.

### FIELDNAME

---

The fieldname appears as a column heading when you include the field in a report request. You can change the default by using AS or NOPRINT in the report or by using TITLE in the Master File.

Fieldnames are unique names of 1-66 characters. The Master File fieldname does not need to be the same as the ADABAS fieldname. Fieldnames can include any alphanumeric characters, but the first character must be a letter from A-Z. Avoid embedded blanks and special characters.

Since all references to data in FOCUS are through fieldnames or aliases, the fieldnames should be unique within a Master File. This requirement is true even when the same ADABAS record is included as two differently structured segments in the same Master File. You must specify different fieldnames in the two segments or FOCUS uses the first one in the Master File unless you qualify which field you want, for example, file.field, segment.field.

## ALIAS

---

The ALIAS for a single ADABAS field *must* contain the two-character internal ADABAS fieldname. The Interface uses it to generate direct calls to the ADABAS DBMS. The format rules are as follows:

- The ALIAS equals the two-character internal ADABAS fieldname, for example, AM.
- The letter C is appended to the two-character internal ADABAS fieldname if the field is the counter field for a PE group or MU field, for example, AMC.
- The ALIAS is the two-character internal ADABAS fieldname appended with the digit 1 if an application calls for only the first occurrence of a repeating field, for example, AM1.
- The ALIAS can be used to reformat the length of the ADABAS field. See *Using the FOCUS ALIAS to Reformat ADABAS Fields* for examples using the reformatting option.

**Note:** Because of the requirement that the ALIAS in a Master File must be the same as the two-character internal ADABAS fieldname, your Master File may include duplicate ALIAS values. In that case, the fieldname is used to differentiate between the segments.

### Using the FOCUS ALIAS to Reformat ADABAS Fields

The FOCUS ALIAS field allows reformatting of the field for a different view of the field. Several ADABAS data formats have maximum lengths that exceed the supported maximum length of FOCUS data formats. You can use the FOCUS ALIAS to convert this data to an acceptable FOCUS ACTUAL format and length.

When a field is specified in a retrieval request, the value in the FOCUS ALIAS is passed to ADABAS in the Format buffer. ADABAS uses the ALIAS to process the field and return the data to FOCUS.

To use the reformatting option, define the ALIAS using length and format values that indicate the way you want to reformat the field.

The syntax is

```
ALIAS= 'xx, l, f'
```

where:

- |           |  |
|-----------|--|
| <i>xx</i> | Is the two-character internal ADABAS fieldname.          |
| <i>l</i>  | Is the new length of the field value in number of bytes. |
| <i>f</i>  | Is the new ADABAS format of the field value.             |

The entire value must be enclosed in single quotation marks.

## Describing ADABAS Files to FOCUS

For example, if you only want to use the first three characters in a six-byte field, use the following syntax in your Master File:

```
FIELD=L_DBNR ,ALIAS='AU,3,P', USAGE=P3, ACTUAL=P3 , $
```

In this case, the ADABAS Interface will read only the first three bytes (L\_DBNR is a six-byte field with the ADABAS description: 01, AU, 6, U).

In the next example, to convert an ADABAS binary field format of ten bytes (ADABAS description: 01, BB, 10, B) to a FOCUS alphanumeric format large enough to accommodate the ADABAS value, you could use the following syntax in your Master File:

```
FIELD=TESTFLD ,ALIAS='BB,10,A', USAGE=A20, ACTUAL=A10 , $
```

Notice that you must also change the USAGE and ACTUAL values to reflect the changes in the format. Refer to the chart in the section on ACTUAL for the acceptable USAGE and ACTUAL values along with the corresponding ADABAS codes.

**Note:** Groups cannot be reformatted. This formatting is only valid for elementary fields.

## USAGE

---

The USAGE attribute enables you to describe how you want fields displayed on the screen, printed in reports, or used in calculations. This attribute includes the data field type, length, and any applicable edit options when the field values are printed.

The syntax is

```
{USAGE}=tllleeeee
```

where:

USAGE                    Indicates how to display a field. FORMAT is a synonym for USAGE.

t                        Is the field type.

lll                      Is the number of positions needed to display the field.

eeee                    Are the edit options.

The value that you specify for field length is the number of alphanumeric characters or numeric digits that FOCUS allocates for the field in any display or report. The specified value excludes the editing characters, such as comma, \$, and so on. Edit options control how a field value is printed.

For more information about the USAGE attribute, refer to the *FOCUS for IBM Mainframe User's Manual*.

## ACTUAL

---

The ACTUAL attribute describes the length and type of the ADABAS field in storage. It is used only to describe the format of fields from external data files, and must be included in the Master Files needed for the ADABAS Interface. The source of this information is your existing ADABAS Field Definition Table (FDT).

The syntax is

```
ACTUAL=tlll
```

where:

*t* Is the FOCUS field type of the ADABAS field.

*lll* Is the storage length of the ADABAS field.

FOCUS permits the following conversions from ACTUAL format to USAGE (display) format:

ADABAS Format Type	FOCUS ACTUAL Type	FOCUS ACTUAL Length*	FOCUS USAGE Type	FOCUS USAGE Length*	Description
A	A	1-256	A	1-256	Alphanumeric
G	D	8	D	1-15	Decimal, Double-Precision
F	F	4	F	1-9	Decimal, Single-Precision
B**	I	1-4	I	1-9	Integer Values
P	P	1-15	P	1-33. <i>n</i>	Packed Decimal
U	Z	1-29	A or P	1-33. <i>n</i>	Zoned

\* Lengths are measured as follows:

FOCUS ACTUAL is in number of bytes.

FOCUS USAGE allows space for all possible digits, a minus sign for negative numbers, and a decimal point in numbers with decimal digits.

\*\* The length for ADABAS format type B (integer values) is 1-126 bytes. This value converts to a FOCUS ACTUAL type of integer I with 1-4 bytes. For example, ADABAS format B126 is represented in FOCUS as I4. Refer to the table containing USAGE and ACTUAL conversion formulas in Chapter 4, *The AUTOADBS Facility*.

The following table shows the codes for the types of data FOCUS reads:

<b>ACTUAL Type</b>	<b>Meaning</b>
A $n$	Alphanumeric characters A-Z, 0-9, and the special characters in the EBCDIC display mode, where $n = 1-256$ bytes.
D8	Double-precision, floating-point numbers, stored internally in eight bytes.
F4	Single-precision, floating-point numbers, stored internally in four bytes.
I $n$	Binary integers, where $n = 1-4$ .
P $n$	Packed decimal data, where $n = 1-15$ bytes.
Z $n$	Zoned decimal data, where $n = 1-30$ bytes. Represent the field as Z $m.n$ , where $m$ is the total number of digits, and $n$ is the number of decimal places (for example, Z6.1 means a six-digit number with one decimal place).

### Example

The following is a typical field declaration:

```
FIELD=MODEL ,ALIAS=AE ,USAGE=A20 ,ACTUAL=A20 ,,$
```

### 5.1.4 Specifying INDEX=I

As stated in Chapter 3, *ADABAS Overview and Mapping Considerations*, descriptors act as key values associated with a single field in an ADABAS record. They enable records to be retrieved more efficiently based on the selection of a value, a set of values, or a range of values.

Additionally, all descriptors, superdescriptors, and subdescriptors are used to establish logical relationships between files. In FOCUS there are two methods for establishing logical relationships:

- The JOIN command to establish the relationship dynamically at runtime. Refer to Chapter 7, *Interface Navigation*, for more information about using ADABAS files that are to be cross-referenced using JOIN.
- A static cross-reference in the Master and Access Files. Section 5.2.5, *Implementing Embedded JOINS: KEYFLD and IXFLD*, discusses how to define shared files.

ADABAS descriptor fields are identified in the Master File using the INDEX=I attribute. For example:

```
FIELD=MAKE ,ALIAS=AD ,USAGE=A20 ,ACTUAL=A20 ,INDEX=I , $
```

INDEX=I is optional in the Master File, but its use is recommended so that keys in your Master File are easily recognized without reference to the Access File.

## 5.1.5 Describing Superdescriptors

---

A superdescriptor is a key value associated with all or part of two to five ADABAS fields. If the superdescriptor consists of a group of complete fields, it is declared differently than a superdescriptor which consists of one or more partial fields. For information on superdescriptors consisting of partial fields, see Section 5.2.4, *Describing Descriptors in the Access File*.

Superdescriptors can be printed if they contain only alphabetic fields.

### Specifying Superdescriptors Using the GROUP Attribute

---

A superdescriptor composed of several complete fields is described with the GROUP attribute. All the fields that are part of the group must be specified immediately after the GROUP attribute in the order in which they appear in the superdescriptor.

The following example shows a superdescriptor defined in FOCUS terminology:

GROUP=TRANS	,ALIAS=S1	,USAGE=A20	,ACTUAL=A20	,INDEX=I	, \$
FIELD=ACCT_NO	,ALIAS=C2	,USAGE=A9	,ACTUAL=A9	, \$	
FIELD=ITEM_NO	,ALIAS=GD	,USAGE=A5	,ACTUAL=A5	, \$	
FIELD=TRANS_DATE	,ALIAS=D7	,USAGE=A6YMD	,ACTUAL=A6	, \$	

**Figure 5-2. Superdescriptor Defined With the GROUP Attribute**

The group fieldname, in this case, is TRANS. Since it is a superdescriptor, the INDEX=I attribute is included on the GROUP specification.

The ALIAS is the two-character ADABAS fieldname for the superdescriptor.

The USAGE and ACTUAL values must be specified in alphanumeric format. See the next section on calculating group length for more information.

If any of the components of the TRANS group field were descriptors, they could also contain the INDEX=I attribute.

**Note:** TYPE=SPR must be defined in the Access File. For more information, see Section 5.2.4, *Describing Descriptors in the Access File*.

## Calculating GROUP Length

---

For a group field, you must supply both the USAGE and ACTUAL values in alphanumeric format.

If the component fields of the group are alphanumeric, the USAGE and ACTUAL lengths of the group field must be exactly the sum of the component field lengths. For example, in Figure 5-2, the lengths specified in the USAGE and ACTUAL attributes for the component fields ACCT\_NO, ITEM\_NO, and TRANS\_DATE each total 20. The lengths specified in the USAGE and ACTUAL attributes of the TRANS group field are consequently both equal to 20.

If the component fields of the group are *not* alphanumeric:

- The USAGE and ACTUAL formats of the group field must still be alphanumeric.
- The ACTUAL length of the group field is still the sum of the component field lengths.
- The USAGE length is the sum of the internal storage lengths of the component fields.

You determine these internal storage lengths as follows:

USAGE	Length
A (alphanumeric)	A value equal to the number of characters contained in the field
D (decimal, double-precision)	8
F (decimal, single precision)	4
I (integer)	4
P (packed decimal): <i>Pn</i> or <i>Pn.d</i> , where <i>n</i> is less than or equal to 15	8
P16	8
P16. <i>d</i>	16
<i>Pn</i> or <i>Pn.d</i> , where <i>n</i> is greater than 16 or less than or equal to 31	16
Z (zoned)	A value equal to the number of characters contained in the field

For example, consider the following GROUP specification in the EMPLOYEES Master File:

GROUP= LEAVE_DATA	,ALIAS= A3	,A16	,A4	, \$
FIELD=LEAVE_DUE	,ALIAS= AU	,P2	,Z2	, \$
FIELD=LEAVE_TAKEN	,ALIAS= AV	,P2	,Z2	, \$

**Figure 5-3. GROUP Attribute in the EMPLOYEES Master File**

In this example:

- The lengths of the ACTUAL values for the component fields LEAVE\_DUE and LEAVE\_TAKEN total 4, which is the length of the ACTUAL value for the group field.
- The length of the USAGE value for the group field is determined by adding the internal storage lengths of the component fields LEAVE\_DUE and LEAVE\_TAKEN, as specified by the field types: a length of 8 for USAGE P2 (8 plus 8 = 16).

Here is a GROUP specification:

GROUP= MODEL_YEAR_MAKE	,ALIAS= AO	,A28	,A22	, INDEX=I, \$
FIELD=YEAR	,ALIAS= AG	,P2	,Z2	, \$
FIELD=MAKE	,ALIAS= AD	,A20	,A20	, INDEX=I, \$

**Figure 5-4. GROUP Attribute in a Sample Master File**

In this example:

- The lengths of the ACTUAL values for the component fields YEAR and MAKE total 22, which is the length for the ACTUAL value for the group field.
- The length of the USAGE value for the group field is determined by adding the internal storage lengths of the component fields YEAR and MAKE, as specified by the field types: a length of 8 for USAGE P2 and a length of 20 for USAGE A20 (8 plus 20 = 28).

## 5.1.6 Describing Multi-value Fields and Periodic Groups With the OCCURS Attribute

The OCCURS attribute is a segment attribute used to describe portions of records containing a multi-value field or a periodic group. When describing ADABAS files, the OCCURS attribute must equal the two-character internal ADABAS name appended with the letter C.

Records with repeating fields are defined to FOCUS by describing the singly occurring fields in the parent segment, and the multiply occurring fields or groups in their own subordinate segment. The OCCURS attribute appears in the declaration for the subordinate segments.

The syntax is

```
OCCURS=occursname , $
```

where:

*occursname* Indicates a data field in the parent segment that contains the number of occurrences of the descendant segment. Its value is derived from the two-character internal ADABAS name (FOCUS ALIAS field) appended with the letter C.

The following is an example of the VEHICLES Master File, containing an MU segment with the counter field in the parent segment:

```
FILENAME=VEHICLES , SUFFIX=ADBSINX , $
SEGNAME=S01 , SEGTYPE=S , $
FIELD= REG_NUM , ALIAS= AA , A15 , A15 , INDEX=I , $
FIELD= CHASSIS_NUM , ALIAS= AB , I9 , I4 , $
FIELD= PERSONNEL_ID , ALIAS= AC , A8 , A8 , INDEX=I , $
FIELD= CLASS , ALIAS= AH , A1 , A1 , INDEX=I , $
FIELD= LEASE_PUR , ALIAS= AI , A1 , A1 , $
FIELD= CURR_CODE , ALIAS= AL , A3 , A3 , $
FIELD= MAINT_COST_CNT , ALIAS= AMC , I4 , I2 , $
GROUP= MODEL_YEAR_MAKE , ALIAS= AO , A28 , A22 , INDEX=I , $
FIELD= YEAR , ALIAS= AG , P2 , Z2 , $
FIELD= MAKE , ALIAS= AD , A20 , A20 , INDEX=I , $
SEGNAME=AM0101 , SEGTYPE=S , PARENT=S01 , OCCURS=AMC , $
FIELD= MAINT_COST , ALIAS= AM , P7 , P4 , $
FIELD= AM0101_OCC , ALIAS= ORDER , I4 , I2 , $
```

*Figure 5-5. Master File With an MU Example*

MAINT\_COST is the MU field. It is allocated to the newly created segment, AM0101. The counter field, MAINT\_COST\_CNT, is located in the parent segment. The new segment, AM0101, uses this count field to determine the number of occurrences of the MAINT\_COST field (OCCURS=AMC).

Notice that the fields MAINT\_COST\_CNT and AM0101\_OCC (ALIAS=ORDER) have USAGE=I4 and ACTUAL=I2. These are the required values for the USAGE and ACTUAL attributes of the counter field and the ORDER field.

The Interface builds the PE segment in the same way. However, the component fields within the PE group are defined in the new segment.

The following is a view of the EMPLOYEES Master File, containing a PE group:

```

FILENAME=EMPFIL1,SUFFIX=ADBSINX,$

SEGNAME=S01      ,SEGTYPE=S,$
FIELD= PERSONNEL_ID           ,ALIAS= AA      ,A8      ,A8      , INDEX=I,$
FIELD=FIRST_NAME             ,ALIAS= AC      ,A20     ,A20     ,$
FIELD=NAME                   ,ALIAS= AE      ,A20     ,A20     , INDEX=I,$
FIELD= LEAVE_BOOKED_CNT      ,ALIAS= AWC     ,I4      ,I2      ,$

SEGNAME=AW0401   ,SEGTYPE=S,PARENT=S01 ,OCCURS=AWC,$
GROUP= LEAVE_BOOKED           ,ALIAS= AW      ,A16     ,A12     ,$
FIELD=LEAVE_START             ,ALIAS= AX      ,P6      ,Z6      ,$
FIELD=LEAVE_END               ,ALIAS= AY      ,P6      ,Z6      ,$
FIELD= AW0401_OCC             ,ALIAS= ORDER   ,I4      ,I2      ,$

```

**Figure 5-6. Master File With a PE Group**

LEAVE\_BOOKED is the PE group. It is allocated to the newly created segment, AW0401. The counter field, LEAVE\_BOOKED\_CNT, is located in the parent segment. The new segment, AW0401, uses this counter field to determine the number of occurrences of the LEAVE\_BOOKED group (OCCURS=AWC).

The counter field, LEAVE\_BOOKED\_CNT, and the ORDER field, AW0401\_OCC, require values of I4 for the USAGE attribute and I2 for the ACTUAL attribute.

**Note:** For both counter (CNT) and ORDER fields, Release 7.0.8 and higher supports ACTUAL=I2. Prior releases required ACTUAL=I1.

The number of occurrences is limited by the original definition of the file to ADABAS.

To retrieve the appropriate occurrence of a repeating field in a report request, specify the ORDER field in your Master File. See Section 5.1.8, *Specifying OCCURS Segment Sequence: The ORDER Field*, for details.

For detailed information about defining MU fields and PE groups in the Access File, see Section 5.2.2, *Segment Attributes*.

Refer to the *FOCUS for IBM Mainframe User's Manual* for more information on the OCCURS attribute.

## 5.1.7 Describing Multi-value Fields Within Periodic Groups

This section describes how to specify multi-value fields within periodic groups in the Master File. Define each multi-value field contained in the periodic group segment as a separate descendant segment of the periodic group.

A periodic group containing a multi-value field is defined as two separate segments.

All fields must be defined in the order in which they appear in the FDT.

Consider the EMPLOYEES file. It has a periodic group called INCOME, which contains one multi-value field, BONUS. Each count field (for example, AQC) must be in its appropriate parent segment. The INCOME\_CNT field is in the parent segment for the periodic group, INCOME. The BONUS\_CNT field is in the parent segment for the MU field, BONUS. Any repeating fields within the periodic group are defined in the AQ0201 segment. Their corresponding ADABAS two-character fieldnames are the values for ALIAS with the appropriate values for USAGE and ACTUAL.

Here is the view of the Master File for this structure:

```

FILENAME=EMPFIL1,SUFFIX=ADBSINX,$
  SEGNAME=S01      ,SEGTYPE=S,$
    FIELD= PERSONNEL_ID      ,ALIAS= AA      ,A8      ,A8      ,
                                INDEX=I,$
    FIELD=FIRST_NAME        ,ALIAS= AC      ,A20     ,A20     ,$
    FIELD=NAME              ,ALIAS= AE      ,A20     ,A20     ,
                                INDEX=I,$
  1. FIELD= INCOME_CNT      ,ALIAS= AQC      ,I4      ,I2      ,$
  1.SEGNAME=AQ0201      ,SEGTYPE=S,PARENT=S01      ,OCCURS=AQC,$
  1.$PEMU = INCOME        ,ALIAS= AQ      ,A19     ,A13     ,$
  1. FIELD=CURR_CODE      ,ALIAS= AR      ,A3      ,A3      ,$
  1. FIELD=SALARY         ,ALIAS= AS      ,P9      ,P5      ,$
  2. FIELD=BONUS_CNT      ,ALIAS= ATC     ,I4      ,I2      ,$
  1. FIELD= AQ0201_OCC    ,ALIAS= ORDER   ,I4      ,I2      ,$
  2.SEGNAME=AT0301      ,SEGTYPE=S,PARENT=AQ0201 ,OCCURS=ATC,$
  2. FIELD= BONUS         ,ALIAS= AT      ,P9      ,P5      ,$
  2. FIELD= AT0301_OCC    ,ALIAS= ORDER   ,I4      ,I2      ,$

```

*Figure 5-7. Master File With an MU Within a PE*

This explanation interprets the file shown in Figure 5-7:

1. PE fields.

This example includes the periodic group called INCOME, which contains the MU field, BONUS. AUTOADBS comments out the group field with \$PEMU because its length is variable (depending on the number of occurrences of the MU field). FOCUS cannot report on a group with an unknown length. However, the component fields can be referenced for reporting purposes.

2. MU fields.

In Figure 5-7, the BONUS\_CNT (ALIAS=ATC) field is the count field. It is defined in the PE segment and tells us the number of occurrences for that field. Used in the MU segment, it indicates how many times this repeating field occurs, for example, OCCURS=ATC. Defined in the parent segment, the count field is used by the child segment to tell us how many occurrences to expect.

Any periodic group segment must be described in its entirety. This description has two requirements: each field of the periodic group must have a field declaration in the periodic group segment, and each field in the group must be described in the order in which it appears in the ADABAS FDT.

### 5.1.8 Specifying OCCURS Segment Sequence: The ORDER Field

---

There may be occasions with OCCURS segment processing when the order of the data is significant. For example, the field values may represent monthly or quarterly data, but the record itself may not explicitly specify the month (or quarter) to which the data applies. The ORDER field maintains the sequence number for each multiply occurring instance.

The order of the occurrences of a multi-value field or periodic group also has some significance in your application. For example, if a periodic group contains 12 occurrences, each occurrence could correspond to one month of the year. The first occurrence represents January, the second February, and so on through to December.

To use the ORDER option, you must include an additional field in your Master File with an ALIAS of ORDER. This option instructs FOCUS to determine the sequence number of fields in an OCCURS segment. FOCUS automatically supplies a value for the new field that defines the sequence number of each repeating group. The ORDER field does not represent an existing ADABAS field; it is used only for internal processing.

The syntax rules for the ORDER field are:

- It must be the last field described in the OCCURS segment.
- The fieldname must be unique.
- The ALIAS must be ORDER.
- The USAGE must be I4, with any appropriate edit options.
- The ACTUAL must be I2 (as of FOCUS Release 7.0.8; prior releases were I1).

Each field with an ALIAS value of ORDER tracks the order of occurrence of the preceding fields in the segment. Therefore, in Figure 5-7, AT0301\_OCC maintains the sequence of BONUS occurrences.

An ORDER field may be decoded into a meaningful value in a DEFINE. Refer to the *FOCUS for IBM Mainframe User's Manual* for information on DEFINE.

## 5.2 Access Files

---

Access Files store database access information used to translate report requests into the required ADABAS direct calls.

In the MVS environment, the Access File is located in a PDS allocated to the ddname FOCADBS or ACCESS. Each member is a separate Access File. In the CMS environment, each Access File is stored with the filetype FOCADBS or ACCESS.

An Access File consists of 80-character records in comma-delimited format. A record in an Access File contains a list of attributes and values, separated by commas and terminated with a comma and dollar sign (,\$). This list is free-form and spans several lines if necessary. You can specify attributes in any order.

FOCUS reads blank lines and lines starting with a dollar sign in column one as comments.

Every Access File contains a release declaration and at least one segment declaration. Release declarations and segment declarations each have their own set of attributes. These attributes are discussed in the following sections.

The following is a sample Access File:

```

$$$ CREATED BY AUTOADBS ON 04/17/95 AT 15.04.22 BY PMSMJ
$$$ FILENAME=EMPFIL1,SUFFIX=ADBSINK,$
RELEASE=6,OPEN=YES,$

$ ADABAS FILE = EMPLOYEES                                DICTIONARY =
SEGNAM=S01 ,ACCESS=ADBS,FILENO=001,DBNO=1,CALLTYPE=RL,
SEQFIELD=PERSONNEL_ID,$
FIELD= DEPT_PERSON                                     ,TYPE=SPR,$
FIELD=DEPT                                             ,TYPE=DSC,NU=YES,$
FIELD=NAME                                             ,TYPE= ,NU=NO,$
FIELD= LEAVE_LEFT                                     ,TYPE=SPR,$
FIELD=LEAVE_DUE                                       ,TYPE= ,NU=YES,$
FIELD=LEAVE_TAKEN                                     ,TYPE= ,NU=YES,$
FIELD= DEPARTMENT                                     ,TYPE=NOP,NU=NO,$
SEGNAM=AI0101,ACCESS=MU ,FILENO=001,$ ADDRESS_LINE
SEGNAM=AQ0201,ACCESS=PE ,FILENO=001,DBNO=1,$ INCOME
SEGNAM=AT0301,ACCESS=MU ,FILENO=001,DBNO=1,$ BONUS
SEGNAM=AW0401,ACCESS=PE ,FILENO=001,$ LEAVE_BOOKED
SEGNAM=AZ0501,ACCESS=MU ,FILENO=001,$ LANG

```

*Figure 5-8. Sample Access File*

## 5.2.1 Release Declaration

The release declaration must be the first uncommented line of the Access File. It identifies the release of ADABAS and indicates whether the ADABAS Interface issues ADABAS OPEN and CLOSE calls for each report request.

The syntax is

```
RELEASE= relnum [,OPEN={YES|NO}] , $
```

where:

<i>relnum</i>	Is the ADABAS release number.
OPEN	Specifies whether the ADABAS Interface issues ADABAS OPEN and CLOSE calls to ADABAS in each report request. Possible values are:
<u>YES</u>	Indicates that ADABAS OPEN and CLOSE calls are issued. This value is the default.
NO	Indicates that ADABAS OPEN and CLOSE calls are not issued.

## RELEASE

---

The first declaration in the Access File contains the ADABAS release number with the attribute RELEASE. Specify the full release number. The value is for documentation only.

The following example illustrates the use of the RELEASE attribute to specify the ADABAS release number:

```
RELEASE=6 , $
```

## OPEN

---

The OPEN attribute specifies whether the ADABAS Interface issues ADABAS OPEN and CLOSE calls to ADABAS in each report request. Specifying YES or NO achieves the following results:

Value	Result
<u>YES</u>	The ADABAS Interface issues an OPEN call to ADABAS at the start of a call set initiated by a retrieval request. An ADABAS CLOSE is issued at the end of retrieval for the request. YES is the default value. <pre>RELEASE=6 , OPEN=YES , \$</pre>
NO	ADABAS OPEN and CLOSE calls are not issued for any retrieval request. <pre>RELEASE=4.1 , OPEN=NO , \$</pre>

Accept the default value (YES) unless you are using an ADABAS release lower than 5.0.

## 5.2.2 Segment Attributes

---

Segment declarations contain detailed information about each segment of an ADABAS file. The segment declaration attributes specify the segment name, the file and database numbers, and the ADABAS password, as well as retrieval options.

The syntax is

```
SEGNAME=segname, ACCESS=access, FILENO=file_number [,DBNO=database_number]
[,CALLTYPE={FIND|RL}] [,SEQFIELD=seqfield] [,PASS=password]
[,FETCH = {ON|OFF}] [,FETCHSIZE = {n|MAX}] , $
```

where:

<i>segname</i>	Is the SEGNAME value from the Master File.
<i>access</i>	Specifies the segment type that the ADABAS Interface uses for each segment. Possible values are: <ul style="list-style-type: none"> <li><u>ADBS</u> For a segment that contains only non-repeating fields. This value is the default.</li> <li><u>PE</u> For a segment that is a periodic group in the same file as the parent segment.</li> <li><u>MU</u> For a segment that is a multi-value field in the same file as the parent segment.</li> </ul>
<i>file_number</i>	Is the number that identifies an ADABAS file.
<i>database_number</i>	Is the number that identifies an ADABAS database.
<u>CALLTYPE</u>	Indicates the type of data retrieval call constructed by the ADABAS Interface. Possible values are: <ul style="list-style-type: none"> <li><u>FIND</u> A FIND call is issued when there are one or more WHERE or IF statements in a retrieval request detected against fields defined with INDEX=I. This is the default value.</li> <li><u>RL</u> A Read Logical (RL) call is issued when there are one or more WHERE or IF statements against fields defined with INDEX=I.</li> </ul> <p><b>Note:</b> See your database administrator for the most efficient CALLTYPE to be used at your site.</p>
<i>seqfield</i>	Provides a default index which controls Read Logical (RL) retrieval when there are no IF or WHERE selection tests.
<i>password</i>	Is the ADABAS password for the file.

## Describing ADABAS Files to FOCUS

<code>FETCH</code>	Indicates whether to use the Fetch feature. Possible values are:
<code>ON</code>	Sets the Fetch feature on for the user session. This is the default value.
<code>OFF</code>	Sets the Fetch feature off for the user session.
	If you include this attribute, you must also include <code>FETCHSIZE</code> (see below).
<code>FETCHSIZE</code>	Sets the number of records per buffer. Possible values are:
<code>n</code>	Is the number of records per buffer (1-999). Ten records per buffer is the default.
<code>MAX</code>	Is automatically calculated by the ADABAS Interface to allow the maximum number of records that fit in a 32K buffer.

**Note:** `FETCH` and `FETCHSIZE` are applicable only to segments described as `ACCESS=ADBS`. `FETCH` and `FETCHSIZE` can be set dynamically to override the Access File settings. See Chapter 8, *Environment Commands*, for more information.

Special attributes that apply to embedded JOINS are explained in Section 5.2.5, *Implementing Embedded JOINS: KEYFLD and IXFLD*. Attributes that apply to superdescriptors and subdescriptors are explained in Section 5.2.4, *Describing Descriptors in the Access File*.

---

## SEGNAM

The `SEGNAM` attribute specifies the name of the segment being described. It is the same name as the `SEGMENT` value in the Master File.

---

## ACCESS

The `ACCESS` attribute specifies the segment type that the ADABAS Interface uses for each segment. The values are:

- `ADBS`, which specifies a segment that contains only non-repeating fields. This value is the default.
- `PE`, which specifies a segment that is a periodic group in the same file as the parent segment.
- `MU`, which specifies a segment that is a multi-value field in the same file as the parent segment.

**Note:** All three access values use the attributes `FILENO` and `DBNO`, as shown in Figure 5-9.

The following is the partial Access File for EMPLOYEEES with the specification of PE and MU fields.

```

$$$ FILENAME=EMPFILE1,SUFFIX=ADBSINX,$
RELEASE=6,OPEN=YES,$

$ ADABAS FILE = EMPLOYEEES                                DICTIONARY =
SEGNAM=S01      ,ACCESS=ADBS,FILENO=001,DBNO=1,CALLTYPE=RL,
                SEQFIELD=PERSONNEL_ID,$
SEGNAM=AQ0201 ,ACCESS=PE  ,FILENO=001,DBNO=1,$ INCOME
SEGNAM=AT0301 ,ACCESS=MU  ,FILENO=001,DBNO=1,$ BONUS
SEGNAM=AW0401 ,ACCESS=PE  ,FILENO=001,DBNO=1,$ LEAVE_BOOKED

```

*Figure 5-9. Access File With PE and MU Fields*

## ACCESS=ADBS

ACCESS=ADBS specifies the main segment of the file. This segment contains only non-repeating fields.

## FILENO

---

The FILENO attribute specifies the ADABAS file number. You need this file number to identify the ADABAS file(s) you wish to access. The valid values are 1-255.

Suppose you are accessing two segments: the first is in the EMPLOYEE file, and the second is in the INSURANCE file. To describe this situation in the Access File, use the SEGNAME and FILENO attributes as illustrated in *DBNO* in Section 5.2.2, *Segment Attributes*.

## DBNO

---

The DBNO attribute specifies the ADABAS database number. It is used when you define files from multiple databases in one Master File. If it is not provided, the DBNO will be read from the DDCARD. If the DDCARD is not allocated, the Interface uses the default value, DBNO=0.

The following example illustrates the use of both the FILENO and DBNO attributes to account for the files being in different databases:

```

SEGNAM=ONE      ,FILENO=1  ,DBNO=1  ,... , $
SEGNAM=PAY      ,FILENO=2  ,DBNO=3  ,... , $

```

## CALLTYPE

---

The CALLTYPE attribute affects how the ADABAS Interface processes WHERE and IF statements on descriptors in report requests and how it retrieves cross-referenced file data from a descendant segment. With CALLTYPE, you optionally specify the type of access to use to process the inverted lists for any given segment. Ask your ADABAS database administrator to advise you when you are selecting a CALLTYPE.

The possible values and their meanings are:

Value	Meaning
FIND	<p>An ADABAS FIND call is issued when there are one or more WHERE or IF statements against fields defined to FOCUS with INDEX=I. The search is on specific values of the field.</p> <p>The ADABAS Interface can generate ADABAS FIND (S1) calls even when non-descriptor fields are used as search criteria. This can occur whenever CALLTYPE=FIND is specified in the Access File, and if you include an IF or WHERE test when referencing a non-descriptor field in your TABLE or TABLEF request.</p> <p>To turn off this feature, see the section in Chapter 7, <i>Interface Navigation</i>, that discusses the optimization of the FIND call using non-descriptor fields.</p>
RL	<p>An ADABAS Read Logical call is issued when there are one or more WHERE or IF statements against fields defined to FOCUS with INDEX=I.</p> <p><b>Note:</b> If you have selected a field defined with TYPE=NOP (for example, subdescriptors), a FIND call is issued even if RL has been specified.</p> <p>The first IF statement in a request that refers to a field with INDEX=I determines the inverted list used for the Read Logical access for the root of the subtree. Refer to Chapter 6, <i>Reporting Considerations</i>, for information on subtrees.</p> <p>When retrieving descendant segments, the ADABAS Interface issues a Read Logical call using the descriptor pointed to by IXFLD. For more information on IXFLD, refer to Section 5.2.5, <i>Implementing Embedded JOINS: KEYFLD and IXFLD</i> and the <i>FOCUS for IBM Mainframe User's Manual</i>.</p> <p>For CALLTYPE=RL, all other selection tests on descriptor fields specified in your report request are applied after the record is returned.</p>

## SEQFIELD

---

The SEQFIELD attribute specifies the field you want to use as the sequencing value:

```
RELEASE=6,OPEN=YES,$
$ ADABAS FILE = EMPLOYEES                                DICTIONARY =
SEGNAM=S01      ,ACCESS=ADBS,FILENO=001, DBNO=1,CALLTYPE=RL,
                SEQFIELD=PERSONNEL_ID,$
```

**Figure 5-10. Access File With the SEQFIELD Attribute**

For more information about SEQFIELD, see Appendix A, *Installation Instructions*.

## PASS

---

The PASS attribute specifies the ADABAS password for the file. It is required if the file is password protected.

You must specify the password for each password-protected file you use. For example:

```
SEGNAM=ONE      ,FILENO=1      ,DBNO=1      ,PASS=ONEXX ,...,$
SEGNAM=PAY      ,FILENO=2      ,DBNO=3      ,PASS=USER2  ,...,$
```

For information about encrypting the Access File to prevent unauthorized viewing of passwords, refer to the *FOCUS for IBM Mainframe User's Manual*.

## FETCH

---

The FETCH attribute indicates whether to use the Fetch feature for segments described as ACCESS=ADBS in the Access File. Valid values are ON (default) or OFF.

## FETCHSIZE

---

The FETCHSIZE attribute sets the number of records per buffer. You can supply a specific number or have the ADABAS Interface automatically calculate the maximum number of records that fit in a 32K buffer.

The following is an example of an Access File that enables the Fetch feature and sets the number of records per buffer to 15.

```
$$$ CREATED BY AUTOADBS ON 03/18/97 AT 15.04.22 BY PMSMJB
$$$ FILENAME=EMPFILE1 , SUFFIX=ADBSINX , $
RELEASE=6 , OPEN=YES , $

$ ADABAS FILE = EMPLOYEES                                DICTIONARY =
SEGNAME=S01 , ACCESS=ADBS , FILENO=001 , CALLTYPE=RL ,
SEQFIELD=PERSONNEL_ID , FETCH=ON , FETCHSIZE=15 , $
```

*Figure 5-11. Access File With the FETCH and FETCHSIZE Attributes*

### 5.2.3 Using the GROUP Attribute to Cross-reference Files

The GROUP attribute can be used to cross-reference ADABAS files if the file you want to search does not contain a descriptor. This cross-referencing can be done *only* if fields within the file you want to search correspond to descriptors in the file you are cross-referencing.

Consider this situation: You have a SALES file which contains salesman information. It also has account information, such as COMPANY\_CODE, ITEM\_NUMBER, and ITEM\_NAME.

Suppose you also have an ACCOUNT file containing information about each company in a salesman's territory. The ACCOUNT file has a superdescriptor consisting of COMP\_CODE, IT\_NUMBER, and IT\_NAME. These fields correspond to the fields specified in the SALES file.

You can create a link between the matching fields in the SALES and ACCOUNT files. To do this, describe the matching fields from the SALES file as a group. Then you can join the group field to the superdescriptor in the ACCOUNTS file. The host file is the file, without a descriptor, containing the held values which must be grouped in order to retrieve related records in the second file. For example:

```
FILE=SALES , SUFFIX=ADBSINX , $
SEGNAME=ACCT_SEG , SEGTYPE=S , $
GROUP=LINKFLDS , ALIAS= , USAGE=A20 , ACTUAL=A20 , $
FIELD=COMPANY_CODE , ALIAS=BA , USAGE=A3 , ACTUAL=A3 , $
FIELD=ITEM_NUMBER , ALIAS=BB , USAGE=A5 , ACTUAL=A5 , $
FIELD=ITEM_NAME , ALIAS=BC , USAGE=A12 , ACTUAL=A12 , $
```

*Figure 5-12. The GROUP Attribute Used to Cross-reference Files*

SALES is the host file which uses the superdescriptor (COMPANY) in the ACCOUNT file for the company-related data. The superdescriptor in the ACCOUNT file, easily recognized by TYPE=SPR in the Access File, is composed of fields that correspond to the dummy group in the host. Looking at Figure 5-12 and Figure 5-13, you see how the fields in the LINKFLDS group in the SALES file relate to the superdescriptor COMPANY in the ACCOUNT file:

```
FILE=ACCOUNT , SUFFIX=ADBSINX , $
  SEGNAME=PROD_SEG , SEGTYPE=S , PARENT=ACCTS , $
    GROUP=COMPANY , ALIAS=S1 , USAGE=A20 , ACTUAL=A20 , INDEX=I , $
      FIELD=COMP_CODE , ALIAS=AA , USAGE=A3 , ACTUAL=A3 , $
      FIELD=IT_NUMBER , ALIAS=AB , USAGE=A5 , ACTUAL=A5 , $
      FIELD=IT_NAME , ALIAS=AC , USAGE=A12 , ACTUAL=A12 , $
```

**Figure 5-13. The GROUP Attribute With a Superdescriptor**

Use the GROUP superdescriptor COMPANY to retrieve data for those values using the JOIN command within a FOCUS procedure. The JOIN command or embedded cross-reference completes the link. In this example, you would join LINKFLDS in SALES to COMPANY in ACCOUNT using the following syntax:

```
JOIN LINKFLDS IN SALES TO ALL COMPANY IN ACCOUNT AS J1
```

## 5.2.4 Describing Descriptors in the Access File

Field suffixes are specified in the Access File to identify descriptors, superdescriptors, and subdescriptors. The syntax is

```
FIELD[NAME]=fieldname, TYPE=fieldsuffix, [NU={YES|NO}] , $
```

where:

<i>fieldname</i>	Is the fieldname or group in the Access File.
<i>fieldsuffix</i>	Indicates the field suffix. Possible values are:
DSC	Indicates a descriptor.
SPR	Indicates a superdescriptor made up of whole fields.
NOP	Indicates a subdescriptor or superdescriptor made up of partial fields.
<i>blank</i>	Indicates non-descriptor fields.

## Describing ADABAS Files to FOCUS

NU	Specifies whether null-suppression is in use. Possible values are:
YES	Indicates that the field is described in the Interface with null-suppression. <b>Note:</b> Descriptors with null values are not stored in inverted lists.
NO	Indicates null-suppression is not used. This is the default value.

### Specifying Superdescriptors Containing Partial Fields

---

If a superdescriptor consists of one or more partial fields, that superdescriptor (field) is defined with TYPE=NOP (non-printable) in the Access File. This type of descriptor can be used in selection tests, but neither it nor any part of it can be printed or displayed unless the field is alphanumeric.

The partial fields that comprise such superdescriptors are stored in ADABAS; there is no facility or necessity for identifying them to FOCUS or the Interface. If you look at an Access File that contains a superdescriptor composed of partial fields, you could recognize it by TYPE=NOP, but you cannot list the partial fields that are its components. To use the Interface, identify such a superdescriptor in the Access File with TYPE=NOP.

Any superdescriptor defined to FOCUS with TYPE=NOP has the following limitation: CALLTYPE=RL is not supported when this field is the IXFLD in an embedded cross-reference or JOIN.

See your Software AG documentation for more information about using superdescriptors containing partial fields.

### Specifying Subdescriptors

---

A subdescriptor consists of a partial field value for which an index has been created in ADABAS. It is described at the field level in the Access File as TYPE=NOP. It can be used in selection tests, but it cannot be printed.

Any subdescriptor defined to FOCUS in the Access File with TYPE=NOP has the following limitation: CALLTYPE=RL is not supported when this field is the IXFLD in an embedded cross-reference or JOIN.

## Specifying Null-suppression

---

To allow the ADABAS Interface to create the most efficient calls, while still maintaining integrity of the answer set, any null-suppressed fields that are components of a superdescriptor must be defined in the Access File. The NU attribute is used to define a null-suppressed field.

An NU field defined as a descriptor is not stored in inverted lists when it contains a null value. Any qualifying descriptor records containing a null value are not recognized by a FIND command that refers to that descriptor.

The same is true for subdescriptors and superdescriptors derived from fields described in the Interface with null-suppression. No entry is made for a subdescriptor if the bytes of the field from which it is derived contain a null value and that field is defined with null-suppression (NU). No entry is made for a superdescriptor if any of the fields from which it is derived is an NU field with a null value.

See your Software AG documentation for more information about null-suppression and how it affects data retrieval.

## 5.2.5 Implementing Embedded JOINS: KEYFLD and IXFLD

---

The KEYFLD and IXFLD attributes identify the common fields for parent/descendant relationships in a multi-segment Master File. These relationships are referred to as embedded JOINS or FOCUS views.

For each descendant segment, the KEYFLD and IXFLD attributes specify the fieldnames of the shared field that implements the embedded JOIN. The parent field supplies the value for cross-referencing; the descendant field contains the corresponding value. The Interface implements the relationship by matching values at run time.

The KEYFLD and IXFLD attributes are only valid for ACCESS=ADBS segments.

Attribute	Meaning
KEYFLD	Refers to the FOCUS fieldname in the parent segment whose value is used to retrieve the child segment. This is also known as the primary key.
IXFLD	Refers to the FOCUS fieldname in the child or cross-referenced segment containing the related data. This is also known as the foreign key.

The value for the KEYFLD attribute is a 1- to 66-character fieldname or alias from the parent segment. The value for the IXFLD attribute is a 1- to 66-character fieldname or alias from the descendant segment.

```

FILENAME=AMKTORDR      , SUFFIX=ADBSINX  , $
SEGNAME=MKTORDER
FIELDNAME  =NMARKET_GRP      , BA      , I3      , I2      , INDEX=I , $
FIELDNAME  =QPRODUCT         , BB      , I3      , I2      , $
FIELDNAME  =QNEEDITM         , BC      , A3      , I2      , $
FIELDNAME  =FBUILD           , BD      , A1      , A1      , $
FIELDNAME  =FK_NPRODUCT      , BE      , A4      , A4      , $
FIELDNAME  =FK_NCUSTOMER    , BF      , I3      , I2      , $
FIELDNAME  =DATEMKTO         , BG      , A8      , A8      , $
FIELDNAME  =DATEFBLD         , BH      , A8      , A8      , $

SEGNAME=CUSTOMER , SEGTYPE=U , PARENT=MKTORDER , $
FIELDNAME  =NCUSTMR_GRP     , AA      , I3      , I2      , INDEX=I , $
FIELDNAME  =NAMECUST         , AB      , A15     , A15     , $
FIELDNAME  =DCUSROAD         , AC      , A20     , A20     , $
FIELDNAME  =DCUSTOWN         , AD      , A20     , A20     , $
    
```

**Figure 5-14. Master File With Embedded JOIN Using KEYFLD and IXFLD**

```

RELEASE=6 , OPEN=YES , $
SEGNAM=MKTORDER , ACCESS=ADBS , FILENO=022 , DBNO=001 , $
SEGNAM=CUSTOMER , ACCESS=ADBS , FILENO=021 , DBNO=001 , CALLTYPE= FIND ,
IXFLD=NCUSTMR_GRP , KEYFLD=FK_NCUSTOMER , $
    
```

**Figure 5-15. Access File With Embedded JOIN Using KEYFLD and IXFLD**

**Note:** Include the pair of attributes in the Access File segment declaration for descendant segments. Do not specify them in the segment declaration for the root segment.

A JOIN can be based on more than one field in the host and cross-referenced logical record types. If the ADABAS file uses multiple fields to establish a relationship or link between logical record types, you can specify concatenated fields in an embedded JOIN. You can also specify multiple fields with the dynamic JOIN command. See Chapter 6, *Reporting Considerations*, for more information.

In the multi-field embedded JOIN, the KEYFLD and IXFLD values consist of a list of their component fields separated by slashes (/). Additional Access File attributes are not required. The syntax is

```

KEYFLD = field1/field2/... ,
IXFLD  = cfield1/cfield2/... , $
    
```

where:

*field1/...* Is a composite of up to 16 key fields from the parent segment. Slashes are required.

*cfield1/...* Is a composite of up to 16 key fields from the descendant segment.

The Interface compares each field pair for the data formats prior to format conversion; it evaluates each field pair with the following rules:

- The cross-referenced field in the embedded JOIN must be an ADABAS descriptor field. The host field can be any field.
- Parent and descendant fields must be real fields.
- All participating fields for either the parent or descendant file must reside in the same segment.
- KEYFLD and IXFLD attributes must specify the same number of fields. If the format of the parent field is longer than the format of the descendant field, its values are truncated. If the format of the parent field is shorter, its values are padded with zeros or blanks.
- The KEYFLD and IXFLD attributes can consist of a maximum of 16 concatenated fields in order of high-to-low significance. You must specify the field order: the order determines how the values will be matched.
- The pair of fields in the KEYFLD/IXFLD attribute do not have to have comparable data formats.

To implement JOINS, the ADABAS DBMS converts the alphanumeric FOCUS field formats to equivalent ADABAS field formats in order to perform the necessary search and match operations. When the ADABAS DBMS returns the answer set of matched values, it also converts the values back to alphanumeric formats. The cross-referenced fields must be descriptor fields.

## 6 Reporting Considerations

---

### Topics:

- File Navigation Techniques
- Selection Considerations
- Using the JOIN Command to Process Multiple Files
- Optimization With Null-suppression for CALLTYPE=RL
- Optimization on Group Fields
- Test on Group Field With Numerics

FOCUS report procedure designers have great flexibility in coding the FOCUS description of the part of the ADABAS structure they want to access. This chapter describes the methods of file traversal that the ADABAS Interface uses to determine the relative advantages of different file descriptions.

### 6.1 File Navigation Techniques

---

When you define specific hierarchical representations of ADABAS structures in Master Files, you specify the order in which you want the ADABAS Interface to retrieve records. This procedure is called navigational logic and is usually part of the application program. Navigation techniques using the JOIN command also work this way.

#### 6.1.1 Referencing Subtrees and Record Retrieval

---

The ADABAS Interface selects where to enter the subtree, called the point-of-entry, and the subsequent navigational processing by analyzing the tree structure defined by your Master File (or JOIN structure) and report request. The Interface determines the smallest subtree that contains all the fields needed for retrieval to produce a report.

The smallest subtree is composed of those segments that contain fields referenced by the request, plus the minimum number of additional segments required to connect all the files used in the request.

The Interface only retrieves records in segments in the referenced subtree. Within the subtree, it retrieves records that contain fields required for the report request or records that are needed to provide the correct links between report fields.

The Interface always enters a database through the root segment of the referenced subtree.

## 6.1.2 FOCUS Segment Retrieval Methodology

---

The ADABAS Interface retrieves segments from top-to-bottom, then left-to-right at each level of the hierarchy. It retrieves all unique descendant segments before any non-unique descendant segments.

This treatment of unique segments is consistent with a basic FOCUS principle: for reporting purposes (though not for updating or file organization), a unique child is logically a direct extension of its parent. This principle is an important factor in selecting a FOCUS structure that properly reflects your ADABAS file. The results of SUM and COUNT operations on fields in child segments may depend on whether they have been declared unique or non-unique. FOCUS also treats missing segments differently, depending on whether the segment is declared unique or non-unique.

## 6.1.3 Missing Unique Segments

---

If a segment is specified as unique (SEGTYPE=U or KU), FOCUS regards it as a logical extension of the parent segment. The ADABAS Interface automatically inserts default values (blanks for alphanumeric fields and zeros for numeric fields) if the unique child segment does not exist. As a result, unique segments are always present.

## 6.1.4 Missing Non-unique Segments

---

If a segment is specified as non-unique (SEGTYPE=S or KM), select one of three options for processing a record without descendant segments. The syntax is

```
SET ALL = all_option
```

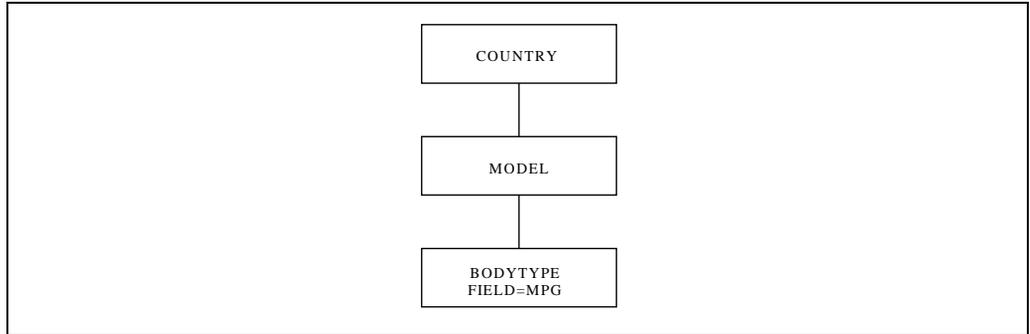
where:

<i>all_option</i>	Allows for the processing of records with no descendant segments. Possible values are:
<u>OFF</u>	Omits parent instances that are missing descendant segments from the report. This value is the default.
ON	Includes parent instances that are missing descendant segments in the report.
PASS	Includes parent instances that are missing descendant segments, even when IF statements exist to screen fields in the descendant segment's missing instances.

Specify SET ALL in your PROFILE FOCEXEC, in any FOCEXEC, or as a FOCUS command.

See the *FOCUS for IBM Mainframe User's Manual* for more information and examples of missing unique and non-unique segments.

The examples in the following sections describing the SET ALL command are based on the following structure:



## SET ALL=OFF

---

The default option (SET ALL=OFF) rejects a record if the request calls for retrieval of a descendant segment and there is no descendant segment present.

For example, assume you have a file in which the parent segment is COUNTRY, which has a descendant segment named MODEL, which in turn has a descendant segment named BODYTYPE. Using the SET ALL=OFF option, the statement

```
COUNT BODYTYPE BY COUNTRY
```

does not print in the report the details of any country that did not produce at least one bodytype of a model of a car.

## SET ALL=ON

---

The ADABAS Interface displays the parent record, even if it has no descendant segments. In this case, using the SET ALL=ON option when processing the statement

```
COUNT BODYTYPE BY COUNTRY
```

FOCUS prints the names of all countries and gives a count of zero (0) bodytypes for those without descendant segments.

## Reporting Considerations

However, if the request has an explicit screening test on the descendant segment, the absence of any descendant segments results in test failure. For example, the request

```
COUNT BODYTYPE BY COUNTRY
IF MPG GT 22
```

excludes any country without any bodytype segments from the report.

## SET ALL=PASS

---

The third option, SET ALL=PASS, allows parents without a descendant segment to pass an explicit screening test on that descendant segment. The request

```
COUNT BODYTYPE BY COUNTRY
```

lists all countries with or without bodytype segments. The request

```
COUNT BODYTYPE BY COUNTRY
IF MPG GT 22
```

includes records without any bodytype segments, and those with an MPG greater than or equal to 22.

## The ALL Prefix

---

Selectively apply SET ALL=ON by adding the ALL prefix to any field from the desired segment.

Reference the field either as a sorting field (for example, BY ALL.COUNTRY or ACROSS ALL.COUNTRY) or as a verb object (WRITE ALL.COUNTRY). As a result, the SET ALL=ON strategy is applied to any missing, immediate, non-unique descendants of the segment containing the ALL-prefixed field. The SET ALL=OFF option is in effect for all other segments.

For example, in the request

```
COUNT MODEL AND BODYTYPE BY ALL.COUNTRY
```

the SET ALL=ON option applies to the country segment and its descendant segments. Therefore, if there is a country without models (and consequently without bodytypes), the report shows that country. Any test condition on either the model or the bodytype segment nullifies the effect of the ALL prefix.

The global SET ALL settings of ON and PASS take precedence over the selective ALL prefix. The selective ALL prefix is effective only when the global setting is OFF, either explicitly or by default.

## 6.2 Selection Considerations

---

The ADABAS Interface analyzes all selection criteria that apply to a specific report request and uses the criteria to minimize its search for data. If a record fails any of the selection tests, FOCUS does not attempt to retrieve any descendant records. Retrieval continues with the next record in the parent segment. If there is no other record in that parent segment and it is not the root of the Master File, FOCUS moves back up to the next record in the previous segment.

The selection tests that you impose on a high-level segment are much more efficient at reducing I/O operations than criteria imposed on lower level segments. If you know in advance which selection criteria are likely to be used most frequently, design the Master File to take advantage of the hierarchical structure in the ADABAS Interface.

### 6.2.1 Selection Order in the Access File

---

When a report request contains multiple optimizable selection tests, the order of descriptors in the Access File determines the order in which FOCUS applies the selection tests. FOCUS issues a Read Logical (RL) call using the first descriptor listed in the Access File that participates in a selection test.

Therefore, for efficient processing you should describe the most restrictive descriptor at the beginning of its segment in the Access File. The order of descriptors in the Master File has no effect on selection processing.

### 6.2.2 RECORDLIMIT and READLIMIT

---

For any request, you may limit the number of ADABAS records retrieved from the database and the number of FOCUS read operations performed. Use the RECORDLIMIT and READLIMIT keywords to impose these limitations by adding the following conditions to a report request

```
{WHERE|IF} RECORDLIMIT {EQ|IS} n
{WHERE|IF} READLIMIT {EQ|IS} n
```

where:

*n* Is the number of records or read operations at which you want to limit the search.

You add these conditions to any report request, or incorporate them into file security through the FOCUS DBA facility. See the *FOCUS for IBM Mainframe User's Manual* for more information on the preceding FOCUS commands and keywords.

## Reporting Considerations

Consider the following example

```
TABLE FILE VEHICLES
PRINT PERSONNEL_ID MAKE MODEL YEAR
WHERE RECORDLIMIT EQ 5
END
```

which produces this report:

PERSONNEL_ID	MAKE	MODEL	YEAR
11500330	CITROEN	BX LEADER	85
11400313	ALFA ROMEO	SPRINT GRAND PRIX	84
30034217	AUSTIN	MINI	80
30034228	TALBOT	SOLARA	83
30008427	AUSTIN	MINI	80

Notice that only five records are reported as requested.

### 6.2.3 Optimization of the FIND Call Using Non-descriptor Fields

It is possible to use non-descriptor fields as search criteria and have the calls to ADABAS use the search buffer rather than read through the entire database.

The ADABAS Interface provides improved optimization by allowing the search buffer to be generated using non-descriptor fields. This optimization occurs whenever `CALLTYPE=``FIND` is specified in the Access File and you include an `IF` or `WHERE` test referencing a non-descriptor field in your report request.

It may prove to be more efficient to alter your retrieval strategy and perform a Read Physical (L2) call when large amounts of data exist. A `SET` command is provided for changing the default ADABAS call when selecting non-descriptor fields.

The syntax is

```
{MVS|CMS} ADBSINX SET NDFIND {ON|OFF}
```

where:

`MVS` Indicates the MVS operating system.

`CMS` Indicates the CMS operating system.

`ON` Causes the search buffer to be generated with any field (non-descriptor and/or descriptor field). This value is the default.

OFF Causes the search buffer to be generated with only descriptor fields. If the request does not use any descriptor field, the Read Physical call is generated.

This command only applies if CALLTYPE=FINN is specified in the Access File.

## 6.3 Using the JOIN Command to Process Multiple Files

---

You can JOIN ADABAS databases defined to FOCUS to other ADABAS databases, to FOCUS files, or to any other fully joinable, FOCUS-readable databases. Each JOIN creates a parent-child relationship. The parent field is called the host or *from* field. The child field is called the cross-referenced or *to* field. See the *FOCUS for IBM Mainframe User's Manual* for more information about the JOIN command.

You can JOIN to ADABAS databases if the cross-referenced field is one of the following:

- A descriptor field.
- A superdescriptor defined with TYPE=SPR or NOP in the Access File.
- A subdescriptor defined with TYPE=NOP in the Access File.

In every case, in the Access File, the cross-referenced segment must specify ACCESS=ADBS.

If CALLTYPE=RL is specified for the cross-referenced segment, the host field can be joined to the high-order portion of a descriptor, superdescriptor, or subdescriptor.

When an ADABAS file is the host file, the host field is one of the following:

- A non-recurring field (ACCESS=ADBS).
- A field in a periodic group (ACCESS=PE).
- A multi-value field (ACCESS=MU).

The ADABAS Interface also supports DEFINE-based JOINS. Up to 16 JOINS can be in effect at one time.

### 6.3.1 Multi-field JOIN and Short-to-long JOIN Capability

For a multi-field JOIN, the number of fields used in the JOIN must be the same for both the host and the cross-referenced files. The cross-referenced fields must describe the left-most portion of a superdescriptor defined to FOCUS using the GROUP attribute. For example:

```
JOIN FLDA AND FLDB IN ADBS1 TO KEY1 AND KEY2 IN ADBS2 AS J1
```

For the short-to-long JOIN, the cross-referenced field must be either a descriptor, subdescriptor, or superdescriptor, or a field that describes the left-most portion of a GROUP superdescriptor.

## 6.4 Optimization With Null-suppression for CALLTYPE=RL

In the ADABAS Interface, optimization refers to using an index to retrieve the answer set. To ensure data integrity and complete answer sets, the ADABAS Interface will perform optimization when:

- All non-referenced component fields of a superdescriptor are *not* null-suppressed (NU=NO in the Access File).
- All component fields of a superdescriptor that contain null-suppressed fields (NU=YES in the Access File) are used in the selection criteria.

If a field is null-suppressed in ADABAS (NU=YES in the Access File), any superdescriptor that uses this field has no entry on the inverted list when this field is blank (alphanumeric) or zero (numeric).

```
GROUP=SUPERG ,ALIAS=S1 ,USAGE=A9 ,ACTUAL=A9 ,INDEX=I , $
FIELD=FLD1 ,ALIAS=AA ,USAGE=A3 ,ACTUAL=A3 , $
FIELD=FLD2 ,ALIAS=AB ,USAGE=A3 ,ACTUAL=A3 , $
FIELD=FLD3 ,ALIAS=AC ,USAGE=A3 ,ACTUAL=A3 , $
```

**Figure 6-1. Master File With a Three-field Superdescriptor**

```
FIELD= SUPERG ,TYPE=SPR, $
FIELD= FLD1 ,TYPE= ,NU=YES , $
FIELD= FLD2 ,TYPE= ,NU=NO , $
FIELD= FLD3 ,TYPE= ,NU=YES , $
```

**Figure 6-2. Access File With a Three-field Superdescriptor**

In order to optimize a selection test against a superdescriptor with null-suppression, you *must explicitly reference the null-suppressed field* in the superdescriptor. If you do not reference the null-suppressed field and the field has no data, there is no record in the index and optimization would return no records. To ensure correct results, FOCUS will not optimize the selection test if the null-suppressed field is not referenced.

See your Software AG documentation for more information about null-suppression and how it affects data retrieval.

## 6.5 Optimization on Group Fields

---

If a report request contains IF or WHERE selection tests against one or more fields that describe the left-most portion of a GROUP descriptor, the ADABAS Interface combines this request into a test that uses the superdescriptor for greater efficiency. If any of the component (or parent) fields of the superdescriptor are defined to ADABAS with null-suppression, be sure to note this information in the Access File to ensure accuracy of reads.

For example, consider the following Master File extract:

```
GROUP=UPERD ,ALIAS=SD ,USAGE=A8 ,ACTUAL=A8 ,INDEX=I , $
  FIELD=PART1 ,ALIAS=P1 ,USAGE=A4 ,ACTUAL=A4 , $
  FIELD=PART2 ,ALIAS=P2 ,USAGE=A4 ,ACTUAL=A4 , $
```

### ***Figure 6-3. Master File With a Two-field Superdescriptor***

If, in a report request, you include the following two tests,

```
WHERE PART1 EQ 'ABCD'
WHERE PART2 EQ 'EFGH'
```

these two tests are equivalent to the following FOCUS syntax:

```
WHERE SUPERD EQ 'ABCD/EFGH'
```

This combination uses the superdescriptor's inverted list and optimizes the ADABAS call. This optimization is performed only if all null-suppressed (NU=YES) superdescriptor components are explicitly referenced in IF or WHERE tests.

## 6.6 Test on Group Field With Numerics

---

If you are testing on a group that contains numeric fields, the test must contain the sign byte. The ADABAS Interface passes only one value per numeric field, based on the preferred sign values in ADABAS. A sign value of F is passed for positive numbers and a sign value of D is passed for negative numbers. This sign value reduces the number of calls sent to ADABAS and also eliminates the need for ADABAS to perform any logical sign translation.

For example:

GROUP=GROUP1	,ALIAS=S1	,USAGE=A9	,ACTUAL=A14	,INDEX=I	, \$
FIELD=FLD1	,ALIAS=AA	,USAGE=A3	,ACTUAL=A3	,	\$
FIELD=FLD2	,ALIAS=AB	,USAGE=P3	,ACTUAL=P3	,	\$
FIELD=FLD3	,ALIAS=AC	,USAGE=A3	,ACTUAL=A3	,	\$

*Figure 6-4. A Group Containing Numeric Fields*

In a report request, you must include the sign for the P3 field:

```
WHERE GROUP1 EQ 'ABC/123F/XYZ'
```

# 7 Interface Navigation

---

## Topics:

- Entry Segment Retrieval of ADABAS Records
- Descendant Periodic Groups and Multi-value Fields
- Descendant ADABAS Records

To retrieve the necessary records that fulfill a request, the ADABAS Interface navigates the ADABAS database through direct calls in read only mode. The calls generated depend on three factors:

- The information supplied in the Master File.
- The information supplied in the Access File for a specified segment.
- The particulars of the report request.

The Interface uses information from these sources to choose the most appropriate and efficient retrieval method.

There are three logical types of ADABAS access:

- Entry segment retrieval of ADABAS records.
- Retrieval of descendant periodic groups and multi-value fields.
- Retrieval of descendant ADABAS records.

This chapter explains the strategies behind each of the above logical types.

## 7.1 Entry Segment Retrieval of ADABAS Records

---

FOCUS constructs an ADABAS request based on the Master File, Access File, and the report request. The root of the subtree is the entry segment into the database for a particular request.

For ADABAS structures, the root of the subtree must be a segment in the Access File containing singly occurring fields (ACCESS=ADBS). The ADABAS calls generated to retrieve data for this entry segment depend on the Access File information that has been supplied for the segment and the selection tests in the request.

The first decision the ADABAS Interface makes for the entry segment is whether to retrieve all the records of the segment or just a subset.

All records are retrieved for a segment if either of the following conditions exists:

- There are *no* selection tests on any field that is a descriptor (defined in the Access File with TYPE=SPR, DSC, or NOP) and the SEQFIELD has not been defined.
- The specified selection test is *not* IF or WHERE and does not use the relational operators:

IS, EQ, GE, GT, LT, LE, or FROM...TO

For example:

```
field IS value1 [(OR value2... OR valuen)]
field IS (ddname)
field FROM value1 TO value2 [(OR value3 TO value4...)]
field GE value1
field GT value1
field LT value1
field LE value1
group EQ 'value1/value2/value3'
```

When you are using one or more of these selection tests against a descriptor, the Interface uses the inverted list to retrieve a subset of the records for the segment.

For example, consider the following selection criterion:

```
field IS value1 [(OR value2... OR valuen)]
```

In this example, any combination of full values or partial values triggers the use of inverted lists except when the \$ (used to indicate a masked field) is in the first position of the value. If the \$ is in the first position of the value, the search conducted by ADABAS is not done through the descriptor's inverted list. Instead, the search is conducted physically through the database in the same manner that a Read Physical call performs a retrieval. See Section 7.1.1, *Read Physical Calls*, for more information, and the *FOCUS for IBM Mainframe User's Manual* for more information on using the \$ as a masking character.

Once the Interface determines whether to retrieve all or a subset of the segment records, it decides which access strategy to use based on the Access File attribute settings. The following are the strategies that the Interface can use:

- Retrieval of all records through Read Physical (L2) calls discussed in Section 7.1.1, *Read Physical Calls*.
- Retrieval of all records through Read Logical (L3) calls without using a starting value discussed in Section 7.1.2, *Read Logical Navigation*.
- Retrieval of a subset of records through Read Logical (L3) calls using a starting value discussed in Section 7.1.2, *Read Logical Navigation*.

- Retrieval of a subset of records through simple FIND (S1) calls discussed in Section 7.1.3, *FIND Navigation*.
- Retrieval of a subset of records through complex FIND (S1) calls discussed in Section 7.1.3, *FIND Navigation*.
- Retrieval of all records through Read Descriptor Value (L9) direct calls discussed in Section 7.1.4, *Read Descriptor Value (L9) Direct Calls*.

The selection logic is designed to implement as many of the record selection tests as possible at the ADABAS level. It minimizes the number of I/O operations required to access the necessary data by issuing efficient calls to ADABAS.

### 7.1.1 Read Physical Calls

---

Read Physical calls read every database record in a physical file, and then return every record for the entry segment to FOCUS. FOCUS must then select the records that meet the request's selection criteria and discard the rest.

The ADABAS Interface issues Read Physical (L2) calls to retrieve all records for the entry segment when:

- The request contains no optimizable selection tests on an inverted list (descriptor) and the Access File does not have a SEQFIELD defined.
- With CALLTYPE=FIND, there is no screening on a descriptor, the screening for a non-descriptor has been set off, and the Access File does not have a SEQFIELD defined.

See Chapter 6, *Reporting Considerations*, for more information about retrieval of records with no screening on a descriptor.

The steps the ADABAS Interface and ADABAS perform to complete a Read Physical call are:

1. The ADABAS Interface constructs a Read Physical (L2) call.
2. ADABAS returns each record in the physical file corresponding to the ADABAS file number specified. The records are retrieved in the order in which they are physically stored.

Subsequent retrieval for the entry segment is completed by issuing the same Read Physical call with the User Control Block unmodified. The ADABAS Interface terminates retrieval when one of the following occurs:

- ADABAS issues a response code indicating end-of-list.
- The FOCUS RECORDLIMIT or READLIMIT is reached (see Chapter 6, *Reporting Considerations*, for information about these keywords).

Read Physical (L2) calls can be faster than Read Logical (L3) calls depending on the request and the data dispersion of the physical storage. Ask your ADABAS database administrator if you should use SEQFIELD for a given ADABAS record, or if Read Physical would be more efficient.

### 7.1.2 Read Logical Navigation

---

When designing a database, it is necessary to know the contents of the files being created. Defining descriptors is very important for efficiency. The ADABAS Interface does not know information about the different fields or which field is more efficient to parse first unless the Access File has this information. The Access File tells the ADABAS Interface how to access your data in the most efficient manner. The order of the fields (for example, DSC, NOP, or SPR descriptors) is very important in the Access File, as the Interface uses this order when executing an IF or WHERE statement.

The Access File needs to have information about the index fields used in the report request. The order in which you define these fields is very important. ADABAS is field oriented, not positional. The order of the fields will not affect retrieval, except for the selection of which inverted list to use.

The Access File controls the order of the IF or WHERE statements, so the database administrator can set up the Access File in the order of the most efficient reads. The user will then receive the requested data much more efficiently. The order in the Access File controls the order of retrieval.

### Read Logical Calls Without a Starting Value

---

The ADABAS Interface uses Read Logical (L3) calls to retrieve all records for the entry segment when the Access File contains a SEQFIELD value for that segment and the report request does not contain an optimizable selection test on an inverted list, and CALLTYPE=RL.

SEQFIELD is generally used to suppress Read Physical calls when there is no optimizable selection test on an inverted list. A Read Physical call will be issued if a SEQFIELD is not defined.

The steps the ADABAS Interface and ADABAS perform to complete a Read Logical call without starting values are:

1. The ADABAS Interface constructs a Read Logical (L3) call without the 'Value Start' option and without the ADABAS Search and Value buffers.
2. ADABAS returns each record corresponding to an entry in the inverted list. The records are in the same order as the inverted list.

Subsequent retrieval for the entry segment is done by issuing the same Read Logical call with the User Control Block unmodified. The records are returned in ascending value of the inverted list. The ADABAS Interface terminates retrieval when one of the following occurs:

- ADABAS issues a response code indicating end-of-list.
- The FOCUS RECORDLIMIT or READLIMIT is reached.

In an ADABAS file that has no record types, you can select the value of SEQFIELD from any inverted list containing entries for all records on the file.

SEQFIELD is required when the ADABAS file has several record types. In this case, the value of SEQFIELD is an inverted list which has entries for a single record type.

If the descriptor used as the SEQFIELD is null-suppressed or contains null-suppressed fields, only records where the descriptor is populated will be included in the retrieval. An entry is not included in the inverted list if a field is null and will not be returned by ADABAS to FOCUS. See your Software AG documentation for more information about null-suppression and how it affects data retrieval.

### Read Logical Calls With a Starting Value

---

The ADABAS Interface constructs Read Logical (L3) calls for the root of the accessed subtree when the following conditions exist:

- The Access File contains CALLTYPE=RL for that segment.
- The request contains an optimizable selection test against a descriptor or superdescriptor identified by TYPE=SPR or DSC in the Access File.
- None of the selection tests are on a field identified by TYPE=NOP in Access File.

When a report request contains multiple optimizable selection tests, the order of descriptors in the Access File determines the order in which FOCUS applies the selection tests. FOCUS issues a Read Logical (RL) call using the first descriptor listed in the Access File that participates in a selection test.

Therefore, for efficient processing you should describe the most restrictive descriptor at the beginning of its segment in the Access File. The order of descriptors in the Master File has no effect on selection processing.

The steps the ADABAS Interface and ADABAS perform to complete a Read Logical call are:

1. The ADABAS Interface constructs a Read Logical (L3) call with the starting value to be retrieved for that inverted list.
2. The ADABAS Interface calls ADABAS with a 'V' in Command Option 2, which instructs ADABAS to use the 'Value Start' option.

3. The call retrieves the first record whose value in the inverted list is equal to or greater than the value specified in the request.

Subsequent retrieval for the entry segment is done by issuing the same Read Logical call with the User Control Block unmodified. The records are returned in ascending value of the inverted list.

The ADABAS Interface terminates retrieval when one of the following occurs:

- The value of the inverted list is out of range of the request.
- ADABAS issues a response code indicating end-of-list.
- The FOCUS RECORDLIMIT or READLIMIT is reached.

If there are multiple values specified in the request (for example, WHERE *field* EQ *value* OR *value*..., WHERE *field* FROM *val1* TO *val2* OR *val3* TO *val4*) and RECORDLIMIT or READLIMIT has not been reached, the ADABAS Interface releases the Command ID and reissues the Read Logical call with the next starting value. For each set of values, the ADABAS Interface terminates retrieval when one of the above occurs.

### 7.1.3 FIND Navigation

---

The manipulation of ISN lists is done on the initial call to ADABAS. Both the intermediate lists and the resulting list may be very large and may take up a large percentage of ADABAS workspace. The ADABAS work area is shared by all ADABAS users, and complex FINDs may affect programs that are updating the database. You may want to suppress FINDs altogether. Accomplish this suppression by specifying CALLTYPE=RL on every Access File segment. Ask your ADABAS database administrator whether to use FIND processing.

#### Simple FIND Calls

---

The ADABAS Interface constructs a simple FIND (S1) call, using a single inverted list structure, for the root of the accessed subtree if one of the following conditions is met:

- The Access File contains either CALLTYPE=FIND for that segment or doesn't specify the CALLTYPE attribute, in which case the Interface default is CALLTYPE=FIND. Additionally, the request contains a single selection test on a descriptor identified with TYPE=DSC or a superdescriptor identified with TYPE=SPR in the Access File.
- For any value of CALLTYPE (RL or FIND), if the request contains a single selection test on a subdescriptor or superdescriptor containing partial fields (identified with TYPE=NOP in the Access File).

CALLTYPE=FIND instructs the Interface to generate FIND calls to retrieve records from the inverted lists. This method is the default for processing selection criteria on inverted lists for a segment (if CALLTYPE is omitted from the Access File segment declaration).

A FIND call is always generated when there is a selection test on a field that has been described in the Access File as TYPE=NOP in a retrieval request.

For performance considerations, a FIND (S1) call does not issue a READ ISN (L1) call if an answer set containing only one record is returned. The GET FIRST option returns the first record in the inverted list. This reduces unnecessary I/O calls. If the FIND call returns more than one ISN in a list, the GET NEXT option of L1 is used to start reading the ISN list from the second entry.

The steps performed by the ADABAS Interface and ADABAS to complete a FIND (S1) call are:

1. The ADABAS Interface constructs a FIND (S1) call with the value(s) specified.
2. The ADABAS Interface calls ADABAS with an 'H' in Command Option 1, which instructs ADABAS to store the resulting list of Internal Sequence Numbers (ISNs) in the ADABAS work area.
3. ADABAS constructs a complete list of ISNs for every record matching the selection criteria in the work area of ADABAS. This list is sorted in ascending ISN order.
4. The ADABAS Interface issues a Read ISN (L1) call to retrieve the first record from the ADABAS work area which matches the selection criteria. The records are returned in ascending ISN order.

All subsequent retrieval for this entry segment is done using the Read ISN (L1) call issued against the ISN list held by ADABAS. L1 commands are issued until one of the following occurs:

- ADABAS issues a response code indicating end-of-list.
- The FOCUS RECORDLIMIT or READLIMIT is reached.

### Complex FIND Calls

---

The ADABAS Interface constructs complex FIND (S1) calls, using two or more inverted list structures, for the root of the accessed subtree if one of the following conditions is met:

- The Access File contains CALLTYPE=FIND for that segment (or omits CALLTYPE). The request contains multiple selection tests on a descriptor or superdescriptor described with TYPE=DSC or TYPE=SPR in the Access File.
- For any value of CALLTYPE, the request contains selection tests on a subdescriptor or superdescriptor containing partial fields (identified with TYPE=NOP in the Access File).

The steps performed by the ADABAS Interface and ADABAS to complete a FIND call on multiple inverted list structures are:

1. The ADABAS Interface constructs a FIND (S1) call with all the values specified for each inverted list on which there is selection criteria.
2. The ADABAS Interface calls ADABAS with an 'H' in Command Option 1, which instructs ADABAS to store the resulting list of ISNs in the ADABAS work area.
3. ADABAS constructs an ISN list for each inverted list specified.
4. ADABAS merges these lists into a final list of ISNs which match all of the selection criteria in the call. This list is kept in the ADABAS work area and is sorted in ascending ISN order.
5. The ADABAS Interface then issues a Read ISN (L1) call to retrieve the first record from the list in the ADABAS work area. The records are returned in ascending ISN order.

All subsequent retrieval for this entry segment is completed using Read ISN (L1) calls issued against the ISN list held by ADABAS. L1 commands are issued until one of the following occurs:

- ADABAS issues a response code indicating end-of-list.
- The FOCUS RECORDLIMIT or READLIMIT is reached.

### 7.1.4 Read Descriptor Value (L9) Direct Calls

---

The ADABAS Interface issues the ADABAS Read Descriptor Value (L9) direct call when you use the FOCUS COUNT verb or SUM CNT *field* within a report request. L9 calls are, by definition, limited to descriptors defined to FOCUS with the following:

- INDEX=I in the Master File.
- A field type of NOP, DSC, or SPR (defined in the Access File).

L9 calls allow an aggregate ISN count to be returned for each unique value on the inverted list at a cost of only one call per unique value. This technique allows a dramatic efficiency gain over passing each record to FOCUS and having FOCUS process the count.

Selection criteria (for example, WHERE *descriptor* EQ '1234') are passed along with the L9 call to further limit the number of calls. If any non-descriptor fields are mentioned in the TABLE request, a Read Descriptor Value (L9) direct call is not possible and FOCUS uses its own internal count processing. If a BY field is used, it must be the same field or GROUP name used as the object of the COUNT verb.

Here is an example of the SUM CNT.*field* using EMPFILE1:

```
TABLE FILE EMPFILE1
SUM CNT.DEPT_S03
BY DEPT_S03
END
```

## 7.2 Descendant Periodic Groups and Multi-value Fields

---

The ADABAS Interface must retrieve a count of the number of occurrences before attempting to retrieve a periodic group (PE) or multi-value (MU) field. This retrieval is accomplished by taking the appropriate ADABAS count field (indicated by the OCCURS attribute on the segment declaration in the Master File) and placing its ALIAS (the two-character ADABAS name of the PE or MU appended with the letter C) in the Format buffer of the call for the parent record fields.

ADABAS returns to this count field the total number of occurrences that exist for the PE or MU.

The ADABAS Interface retrieves the descendant data in one of the following ways:

- For MU fields or PE groups which do not contain an MU, all the occurrences are retrieved at once. The count field ALIAS is used without the letter C, to retrieve the entire set of occurrences in one Read ISN (L2) call.
- For PE groups which contain one or more MU fields, the Interface retrieves a single occurrence of all component fields at a time, including the count(s) of the number of MU descendants, if required. For example, a PE group containing an MU field which has five occurrences of the PE requires five Read ISN (L2) calls to retrieve all of them.

## 7.3 Descendant ADABAS Records

---

The access strategy used to retrieve descendant records in a subtree depends on the Access File attributes specified for a given segment and the SEGTYPE attribute in the Master File (or the SEGTYPE created on the cross-referenced segment as the result of a JOIN). In some cases, the selection criteria in a request further affect access strategy for descendant segments.

Descendant records are related to their parents by a field or GROUP in the parent that corresponds to a field or GROUP in the descendant. This relationship is set up either in the Access File using the KEYFLD/IXFLD pair or through the host field/cross-referenced field combination in a JOIN.

The most appropriate access strategy is selected based on the value of CALLTYPE of the descendent segment in the Access File. The three basic strategies used to obtain descendant records are:

- Retrieval of descendant records through Read Logical calls using a starting value.
- Retrieval of descendant records through simple FIND calls.
- Retrieval of descendant records through complex FIND calls.

The CALLTYPE attribute, together with the KEYFLD/IXFLD attributes, the SEGTYPE, and the TABLE request, are the determining factors in the way descendant data is retrieved.

### 7.3.1 Read Logical Calls Using a Starting Value

---

The ADABAS Interface constructs Read Logical (L3) calls for related descendant records of a parent when the following conditions are met:

- The Access File contains CALLTYPE=RL for that segment.
- The IXFLD or JOIN *to* field is described as TYPE=DSC or TYPE=SPR in the Access File.
- No other selection criteria on inverted lists for this segment have selection on a descriptor with TYPE=NOP.

Read Logical processing is performed on a single inverted list only. The steps the ADABAS Interface performs to complete a Read Logical call using a starting value are:

1. The ADABAS Interface constructs a Read Logical (L3) call with the value of the KEYFLD from the parent segment for the IXFLD inverted list. This call retrieves the first record whose value in the inverted list is equal to or greater than the value of the KEYFLD.
2. For unique descendants, no additional retrieval is performed on the descendant segment for any given parent. (For an embedded cross-referenced segment, the SEGTYPE is U or KU or the segment was the cross-referenced segment in a unique JOIN.)
3. For non-unique descendants, subsequent retrieval for the entry segment is completed by issuing the same Read Logical call with the User Control Block unmodified. The records are returned in ascending value of the inverted list.

The Interface terminates retrieval when one of the following conditions is met:

- The value of the inverted list is greater than the value of the KEYFLD.
- ADABAS issues a response code indicating end-of-list.
- The FOCUS RECORDLIMIT or READLIMIT is reached.

## 7.3.2 Simple FIND Calls

---

The ADABAS Interface constructs a simple FIND (S1) call, using a single inverted list, to obtain related descendant records for a parent when one of the following conditions exists:

- The Access File contains CALLTYPE=FIND (or omits CALLTYPE) and the request contains no selection criteria on the descendant segment.
- The Access File contains CALLTYPE=FIND (or omits CALLTYPE) and the request contains selection criteria on the descendant segment and the descendant is unique. (For an embedded cross-referenced segment, the SEGTYPE is U or KU or the segment was the cross-referenced segment in a unique JOIN.)
- For any value of CALLTYPE, IXFLD is described with TYPE=NOP in the Access File and the request contains no selection criteria on the descendant segment.

The steps the ADABAS Interface and ADABAS perform to complete simple FIND calls against descendant segments are:

1. The ADABAS Interface constructs a FIND (S1) call with the value of the KEYFLD from the parent segment for the inverted IXFLD list.
2. For unique descendants, ADABAS returns the record on the FIND call; no 'H' command is issued to ADABAS. Only one call will be issued, and only the first instance of data will be returned.
3. For non-unique descendants, the ADABAS Interface calls ADABAS with an 'H' in Command Option 1, which instructs ADABAS to store the resulting list of ISNs in the ADABAS work area.
4. ADABAS constructs a complete list of ISNs for every record related to the parent record in the work area. This list is sorted in ascending ISN order.
5. The ADABAS Interface issues a Read ISN (L1) call to retrieve the first record from the ADABAS work area which matches the selection criteria.

If the descendant is not unique, subsequent retrieval of records for this descendant segment is completed using the Read ISN (L1) call issued against the ISN list held by ADABAS. L1 calls are issued until one of the following occurs:

- ADABAS issues a response code indicating end-of-list.
- The FOCUS RECORDLIMIT or READLIMIT is reached.

### 7.3.3 Complex FIND Calls

---

The ADABAS Interface constructs a complex FIND (S1) call, using several inverted list structures, to obtain related descendant records for a parent when one of the following conditions is met:

- The Access File contains CALLTYPE= FIND for that segment (or omits CALLTYPE) and the descendant is non-unique. (For an embedded cross-referenced segment, the SEGTYPE is S or KM, or the segment was the cross-referenced segment in a non-unique JOIN.) The report request contains one or more selection criteria on the descendant segment.
- The Access File contains any value of CALLTYPE. The descendant is non-unique. (For an embedded cross-referenced segment, the SEGTYPE is S or KM, or the segment was the cross-referenced segment in a non-unique JOIN.) The request contains one or more selection criteria on the descendant segment, one of which is on a descriptor, or the IXFLD is described as TYPE= NOP in the Access File.

The steps the ADABAS Interface and ADABAS perform to complete a complex FIND call are:

1. The ADABAS Interface constructs a FIND (S1) call with the value of the KEYFLD from the parent segment using the inverted IXFLD list, in addition to all other selection criteria on inverted lists from the request.
2. For non-unique descendants, the ADABAS Interface calls ADABAS with an 'H' in Command Option 1, which instructs ADABAS to store the resulting list of ISNs in the ADABAS work area.
3. ADABAS constructs an ISN list for each inverted list specified and merges these lists into a final list of ISNs which match all of the selection criteria in the call. This list is kept in the ADABAS work area and is sorted in ascending ISN order.
4. The ADABAS Interface then issues a Read ISN (L1) call to retrieve the first record from the ADABAS work area which matches the selection criteria.

Subsequent retrieval of records for this descendant segment is done through the Read ISN (L1) call issued against the ISN list held by ADABAS. L1 commands are issued until one of the following occurs:

- ADABAS issues a response code indicating end-of-list.
- The FOCUS RECORDLIMIT or READLIMIT is reached.

# 8 Environment Commands

---

## Topics:

- Multifetch and Prefetch Options
- ADABAS Dynamic Database Number
- Overriding Default Passwords in Specific Files
- Running in 24-bit Mode
- Optimization of the FIND Call Using Non-descriptor Fields

This chapter describes ADABAS Interface environment commands that display and change the parameters that govern your FOCUS session.

## 8.1 Multifetch and Prefetch Options

---

The ADABAS Multifetch and Prefetch options reduce execution time and allow ADABAS data to be retrieved with a high degree of efficiency. By buffering multiple record results from a single call, Multifetch and Prefetch reduce the communication overhead between the ADABAS Interface and the ADABAS nucleus.

ADABAS Release 5.3.2 is the first release to support Multifetch, and Release 4.0 is the first release to support Prefetch. The ADABAS Interface uses the Multifetch option if it is available (this is the default option), or the Prefetch option if it is available. The option available is determined by your site's environment.

The Interface trace file FSTRACE4 contains the following information about the Fetch feature:

- Fetch is allowed or disallowed.
- Fetch technique being used (Multifetch or Prefetch).
- Number of records per Fetch buffer.
- Size of the Fetch buffer (ISN buffer).

For more information on FSTRACE4, see Chapter 9, *Debugging Techniques*.

## 8.1.1 Invoking Multifetch/Prefetch

---

The Multifetch/Prefetch feature requires that OPEN=YES (the default value) is specified in the Access File. The Multifetch/Prefetch feature is activated as long as OPEN=YES is specified. There are two ways to deactivate (or reactivate) this feature:

- Issue SET commands before running the request.
- Include the FETCH and FETCHSIZE attributes in the Access File.

**Note:** The SET commands override the Access File settings. This setting will be in effect for the entire FOCUS session.

### SET Command Syntax

---

The syntax for the SET commands is

```
{CMS|MVS} ADBSINX SET FETCH ON
{CMS|MVS} ADBSINX SET FETCH OFF
{CMS|MVS} ADBSINX SET FETCH DEFAULT
{CMS|MVS} ADBSINX SET FETCHSIZE n
{CMS|MVS} ADBSINX SET FETCHSIZE MAX[IMUM]
```

where:

CMS	Indicates the CMS operating system.
MVS	Indicates the MVS operating system.
ON	Sets the Fetch feature on for the user session.
OFF	Sets the Fetch feature off for the user session.
<u>DEFAULT</u>	Uses the information from the Access File. This value is the default.
<i>n</i>	Is the number of records per buffer (1-999). The buffer size varies with the size of the record and can be different for each TABLE request. The buffer size limit is 32K. The default is 10 records per buffer.
MAX	Is automatically calculated by the ADABAS Interface to allow the maximum number of records that fit in a 32K buffer.

### Access File Syntax

---

FETCH and FETCHSIZE are applicable only to segments described as ACCESS=ADBS in the Access File.

The Access File can contain the following attributes in the SEGNAM statement

```
FETCH = {ON|OFF}
FETCHSIZE = {n|MAX[IMUM]}
```

where:

ON	Sets the Fetch feature on for the user session. This value is the default.
OFF	Sets the Fetch feature off for the user session.
n	Is the number of records per buffer (1-999). The default is 10 records per buffer.
MAX	Is automatically calculated by the ADABAS Interface to allow the maximum number of records that fit in a 32K buffer.

Here is an example of the FETCH and FETCHSIZE attributes in the sample Access File for EMPLOYEES:

```
$$$ CREATED BY AUTOADBS ON 03/18/97 AT 15.04.22 BY PMSMJB
$$$ FILENAME=EMPFIL1 , SUFFIX=ADBSINX , $
RELEASE=6 , OPEN=YES , $

$ ADABAS FILE = EMPLOYEES                                DICTIONARY =
SEGNAM=S01 , ACCESS=ADBS , FILENO=001 , CALLTYPE=RL ,
          SEQFIELD=PERSONNEL_ID , FETCH=ON , FETCHSIZE=15 , $
FIELD= DEPT_PERSON          , TYPE=SPR , $
FIELD=DEPT_S03              , TYPE=DSC , NU=YES , $
FIELD=NAME_S03              , TYPE=      , NU=NO , $
```

*Figure 8-1. The FETCH and FETCHSIZE Attributes*

See your Software AG documentation for more information about the Multifetch/Prefetch feature.

## 8.2 ADABAS Dynamic Database Number

---

Previous versions of the ADABAS Interface required specification of the ADABAS database number in the Access File using the DBNO attribute. This feature required a duplicate copy of the Access File for each database accessed.

ADABAS database numbers can now be set from the command level in FOCUS. A new SET command allows users to override the DBNO in the Access File. Note that specifying database numbers in the Access File is still supported. Use of the dynamic database number makes Master and Access Files shareable among databases.

The SET command remains valid throughout the user's session.

## Environment Commands

The syntax is

```
{CMS|MVS} ADBSINX SET DBNO dbno
{CMS|MVS} ADBSINX SET DBNO dbno AFD afd
{CMS|MVS} ADBSINX SET DBNO dbno AFD afd SEG[NAM] segname
{CMS|MVS} ADBSINX SET ?
{CMS|MVS} ADBSINX SET DBNO DEFAULT
```

where:

CMS	Indicates the CMS operating system.
MVS	Indicates the MVS operating system.
<i>dbno</i>	Is any valid numeric database value between 0 and 255.
<i>afd</i>	Is any valid Access filename.
<i>segname</i>	Is any valid ADBS segname in the Access File.
?	Queries the current settings.
DEFAULT	Returns to the default settings for all previous settings. The DBNO is read from the Access File. If the attributes are not specified in the Access File, the Interface will determine the DBNO from the DDCARD.

Examples for MVS are shown below:

MVS ADBSINX SET DBNO 5	Retrieves all data from database number 5.
MVS ADBSINX SET DBNO 2 AFD EMPFILE	Retrieves data for the EMPFILE from database number 2.

**Note:** Several commands can be issued for different files. Each command must be issued separately. Changes are cumulative. For example:

MVS ADBSINX SET DBNO 1 AFD EMPFILE	
MVS ADBSINX SET DBNO 2 AFD EMP2	
MVS ADBSINX SET DBNO 3 AFD VEHICLES	
MVS ADBSINX SET DBNO 2 AFD EMPFILE SEG S02	Retrieves data for the EMPFILE in database number 2 for a particular ADBS segment (segment S02).
MVS ADBSINX SET ?	Displays your settings.
MVS ADBSINX SET DBNO DEFAULT	Returns to the default settings. This resets all SET DBNO commands for all Access Files.

## 8.3 Overriding Default Passwords in Specific Files

---

Passwords for ADABAS files can be set from the command line in FOCUS using the SET PASSWORD command. You can set default passwords for all files and/or databases. Specific passwords can be set for specific files which will override the default. The SET command overrides the password coded in the Access File.

The SET PASSWORD command remains valid throughout the user's session.

The syntax is

```
{CMS|MVS} ADBSINX SET PASSWORD password FILENO ALL DBNO {ALL|dbno}
{CMS|MVS} ADBSINX SET PASSWORD password FILENO fileno DBNO dbno
{CMS|MVS} ADBSINX SET PASSWORD OFF
{CMS|MVS} ADBSINX SET PASSWORD DEFAULT
```

where:

CMS	Indicates the CMS operating system.
MVS	Indicates the MVS operating system.
<i>password</i>	Is the password, which can be from one to eight characters in length.
FILENO	Specifies the file number for which the password is set.
DBNO	Specifies the database or databases for which the password is set.
ALL	Indicates all files and/or databases used with the FILENO and DBNO parameters. If you want to use ALL for both FILENO and DBNO, issue this command before any other subsequent password commands. ALL overrides any prior settings.
<i>dbno</i>	Is any valid numeric database value between 0 and 255. The DBNO parameter indicates the actual database number.
<i>fileno</i>	Provides a file number, a list of file numbers, and/or a range of file numbers used with the FILENO parameter. Numbers and ranges can be combined by separating items with commas. Valid file numbers range between 0 and 255.
OFF	Clears all previous ADBSINX SET PASSWORD commands. The ADABAS Interface does not use any passwords specified in the Access File. This command lets you access only those files that have no password security.
DEFAULT	Clears all previous ADBSINX SET PASSWORD commands and causes the ADABAS Interface to use the password in the Access File.

## Environment Commands

To set a default password:

- Issue the SET PASSWORD command using the ALL keyword for FILENO and/or DBNO.
- Issue specific password requests by specifying particular file and/or database numbers (or ranges) in a subsequent SET PASSWORD command.

Note that subsequent requests are appended to previous requests. Changes are cumulative. The passwords can be issued from a FOCEXEC, which may be encrypted. Refer to the *FOCUS for IBM Mainframe User's Manual* for more information on encryption.

## Examples

---

To set a default password to MARY for all databases and all files under CMS, issue the following command:

```
CMS ADBSINX SET PASSWORD MARY FILENO ALL DBNO ALL
```

To set the password to DAN for all files in database number 150 under CMS, issue the following command:

```
CMS ADBSINX SET PASSWORD DAN FILENO ALL DBNO 150
```

To set the password to BRUCE for specific files in database number 123 under MVS, issue the following command:

```
MVS ADBSINX SET PASSWORD BRUCE FILENO 1,3-5,23,89-93 DBNO 123
```

To clear all previous ADBSINX SET PASSWORD commands under CMS, only allowing the user to access the files that have no password security, issue the following command:

```
CMS ADBSINX SET PASSWORD OFF
```

To clear all previous ADBSINX SET PASSWORD commands under MVS, causing the ADABAS Interface to use the password in the Access File, issue the following command:

```
MVS ADBSINX SET PASSWORD DEFAULT
```

## 8.4 Running in 24-bit Mode

---

The following command is used to allocate buffers below the line because of certain Software AG user exits that must run below the line. Issue this command in your FOCUS session:

```
MVS ADBSINX SET AMODE 24
```

## 8.5 Optimization of the FIND Call Using Non-descriptor Fields

---

It is possible to use non-descriptor fields as search criteria and have the calls to ADABAS use the search buffer rather than read through the entire database.

The ADABAS Interface provides improved optimization by allowing the search buffer to be generated using non-descriptor fields. This optimization occurs whenever CALLTYPE= FIND is specified in the Access File and you include an IF or WHERE test referencing a non-descriptor field in your report request.

It may prove to be more efficient to alter your retrieval strategy and perform a Read Physical call when large amounts of data exist. A SET command is provided for changing the default ADABAS call when selecting non-descriptor fields.

The syntax is

```
{MVS|CMS} ADBSINX SET NDFIND {ON|OFF}
```

where:

MVS	Indicates the MVS operating system.
CMS	Indicates the CMS operating system.
<u>ON</u>	Causes the search buffer to be generated with any field (non-descriptor and/or descriptor field). This value is the default.
OFF	Causes the search buffer to be generated with only descriptor fields. If the request does not use any descriptor field, the Read Physical call is generated.

This command only applies if CALLTYPE= FIND is specified in the Access File.

## 9 Debugging Techniques

---

### Topics:

- Common Errors and Response Codes
- Using Traces
- Verifying the MVS Environment
- Verifying the CMS Environment
- Verification of Installation and AUTOADBS
- Follow-up

From time to time you may experience problems with your ADABAS Interface. In order to address your Interface issues and concerns specifically, please check for the following conditions:

- Can you reproduce your problem outside of ADABAS in a fixed or FOCUS file?
- Did you call the Interface? If there were error messages in the 44xx through 45xx series, an Interface error was produced. What is the response code?
- If no error messages were produced, ensure that the ADABAS.DATA library for MVS is allocated correctly to ddname ERRORS. This library contains the member FOCADA, which lists the Interface error messages. For CMS, be sure that the FOCADA ERRORS file is accessible.

## 9.1 Common Errors and Response Codes

---

The following chart lists common FOCUS error messages and ADABAS response codes.

<b>FOCUS Error</b>	<b>ADABAS Response Code</b>	<b>Description</b>
FOC4490	148	The database is down or unavailable.
FOC4496	148	Check the DBNO in the Access File.
FOC4500	113	The FILENO for the segment in the Access File is not correct.
FOC4504	0	Check the DBNO and FILENO in both the PREDDDB and PREDEL Access Files.
FOC4504	17	The database contains no data and/or the database does not exist in the Predict dictionary.
FOC4504	28	Check the compatibility of the Predict dictionary and FDT listing.
FOC4504	41	The Predict dictionary and FDT may not match. Check the group in the FDT. It may have been created differently in the Predict dictionary.
FOC4504	201	The password is not correct.

See your Software AG documentation for additional response codes. See Appendix B, *Interface Error Messages*, for a complete list of FOCUS error messages.

## 9.2 Using Traces

---

You use the ADABAS Interface trace facility to display the ADABAS Communications buffer generated by the Interface for each call made to ADABAS. This feature is useful for debugging purposes and performance monitoring. It enables you and the database administrator to track ADABAS resource usage from FOCUS. The ADABAS Format, Search, and Value buffers are also displayed when a specific ADABAS call requires one or more of them (refer to your Software AG documentation for more information on buffers).

The following trace options are available for the ADABAS Interface:

- FSTRACE is a very detailed trace that allows you to see exactly what is being passed in the ADABAS buffers. For example, if the Fetch feature is being used, a P displays under the COP1 column if Prefetch is active, or an M appears in that column if Multifetch is active. The COP1 field will be blank if the Fetch feature is not being used.
- FSTRACE4 is a summary trace with ADABAS statistical information. It displays the type of call issued (for example, RL or FIND), the key field used in the search buffer (if applicable), and information about the Fetch feature. The trace options provide this information:
  - Number of I/O operations.
  - Number of commands issued.
  - Amount of CPU time in seconds.
- FSTRACE5 is a trace used to show the flow of control in the Interface.

All trace information is returned after the ADABAS CLOSE call is issued. To generate this information, you must specify OPEN=YES in the Access File.

The following report request uses the EMPLOYEES Master and Access Files. These files are shown in Appendix C, *Sample File Descriptions*. The request produces the report shown. This request generates the FSTRACE illustrated in Figure 9-1, the FSTRACE4 illustrated in Figure 9-2, and the FSTRACE5 illustrated in Figure 9-3.

```
TABLE FILE EMPFILE1
PRINT NAME CITY
WHERE CITY EQ 'NEW YORK'
END
```

NAME	CITY
----	----
RUBIN	NEW YORK
WALLACE	NEW YORK

END OF REPORT

The same report request generates the following trace:

```

*** TRACE START ***=====
1. HID CMD   CID   FNR   RSP  ISN  ISL   ISQ  FBL  RBL  SBL  VBL  IBL
   A  OP  0000  000000  0   0   0    0   0   8   0   0   0
   COP1 COP2 ADD1   ADD2 ADD3   ADD4   CMDT      UA
          *****          *****          159700304 0
   RCD BUFFER: 8
   C1C3C37E F0F0F14B `          *ACC=001.          *
2. HID CMD   CID   FNR   RSP ISN  ISL   ISQ  FBL  RBL  SBL  VBL  IBL
   A  OP  0000  000000  0   0  67108864 100729600  0   8   0   0
   COP1 COP2 ADD1   ADD2 ADD3   ADD4   CMDT      UA
          *****          *****          /` 159700304 e
=====
   HID CMD   CID   FNR   RSP  ISN  ISL   ISQ  FBL  RBL  SBL  VBL  IBL
      L3  0001  000001  0   0   10    0   6  400  3  20  164
3. COP1 COP2 ADD1   ADD2 ADD3   ADD4   CMDT      UA
   M V  AJ          *****          159702136 0
4. FMT BUFFER: 6
   AE,AJ.
5. VAL BUFFER: 20
   D5C5E640 E8D6D9D2 40404040 40404040          *NEW YORK          *
   40404040          *          *
6. SRCH BUFFER: 3
   AJ.
   HID CMD   CID   FNR   RSP ISN  ISL   ISQ  FBL  RBL  SBL  VBL  IBL
      L3  0001  000001  0  563  10    0   6  400  3  20  164
   COP1 COP2 ADD1   ADD2 ADD3   ADD4   CMDT      UA
   M V  AJ]è          *****          /` 159702136 e6
7. RCD BUFFER: 400
   D9E4C2C9 D5404040 40404040 40404040          *RUBIN          *
   40404040 D5C5E640 E8D6D9D2 40404040          *   NEW YORK          *
   40404040 40404040 E6C1D3D3 C1C3C540          *           WALLACE *
          .
          .
          .

```

**Figure 9-1. FSTRACE**

The first section (Item 1) displays commands sent to the ADABAS database. The second section (Item 2) is the ADABAS response.

Certain fields in a trace are particularly helpful in debugging; in the sample trace in Figure 9-1, those fields appear in boldface type. To understand the entire trace, please check your Software AG documentation to interpret the information in the buffers.

The following items provide a sample of the information you need to check:

1. The CMD field shows the command being issued. The first call in this trace is the OP (open) call.
2. The RSP field shows the response code from this call. In the example, the response code is 0.
3. The COP1 field indicates if the Fetch feature is in effect. In the example, the M indicates that Multifetch is active.  
The COP2 field indicates if a starting value is used with the L3 (Read Logical) command. In the example, the V indicates that a starting value is in use.
4. The FMT BUFFER shows the fields used in this request. The AE (NAME) and AJ (CITY) fields are requested.
5. The VAL BUFFER shows the value being searched (NEW YORK).
6. The SRCH BUFFER shows the ADABAS name used for the search (AJ).
7. The RCD BUFFER shows the records returned to the ADABAS Interface from the ADABAS database.

FSTRACE4 shows an abbreviated version of this information using the RL call.

```

data base OPEN mode is ON
access mode for sgx 1 is 4 (RL SCREEN)
format buffer len is 6
AE,AJ.
search buffer len is 3
AJ.
value buffer len is 20
record buffer len is 40
using FETCH technique is ALLOWED
used FETCH technique is MULTIFETCH
number of records per FETCH buffer is 10
FETCH buffer (ISN buffer) size is      164
FETCH buffer (RB buffer) size is      400

ADABAS statistics for DBNO 0
Number of I/O operations : 7
Number of commands issued : 3
Amount of CPU time in sec : 0

```

**Figure 9-2. FSTRACE4**

FSTRACE5 tracks the flow of control in the ADABAS Interface for the same report request. This trace is used infrequently. IBI will request this trace only when necessary.

```
Start scanning the list of "base" files.
Opened AFD EMPFILE1 FOCADBS
  Record type 0000 read ...
  Record type 0000 read ...
  Record type 0001 read ...
  Record type 0000 read ...
  Record type 0002 read ...
    Describing segment S01          ; llevel index 1
  Record type 0003 read ...(6 repetitions of this line follow)
  Record type 0000 read ...
  Record type 0002 read ...(4 repetitions of this line follow)
Closed AFD EMPFILE1 FOCADBS
All segs are done
  *** TRACE START ***adaccx db 0
Adding db 0 to addb
adaccx fl 1
Adding fl 1 to addb
naddb 1
addbfetch db 0 file 1 fetch_active M
addbfetch fetch_size 10 IB_size 164 RB_size 400
Calling adgtseg for sgx 1
padaccx for sgx 1 is 984dbfc
accmmod for sgx 1 is 4
Calling adgtrlscr on sgx 1
bufoff for sgx 1 is 0
buffer for sgx 1 for RDE is 9865010 , for data is 98650d0
GNTINT buffer for sgx 1 is 925e000
Called adgtseg for sgx 1 rc 0 data 1
Calling adgtseg for sgx 1
padaccx for sgx 1 is 984dbfc
accmmod for sgx 1 is 4
Calling adgtrlscr on sgx 1
bufoff for sgx 1 is 0
buffer for sgx 1 for RDE is 9865024 , for data is 98650f8
GNTINT buffer for sgx 1 is 925e000
Called adgtseg for sgx 1 rc 0 data 1
Calling adgtseg for sgx 1
padaccx for sgx 1 is 984dbfc
accmmod for sgx 1 is 4
Calling adgtrlscr on sgx 1
bufoff for sgx 1 is 0
buffer for sgx 1 for RDE is 9865034 , for data is 9865120
GNTINT buffer for sgx 1 is 925e000
Called adgtseg for sgx 1 rc 0 data 0
```

**Figure 9-3. FSTRACE5**

## 9.2.1 Invoking the Trace Facility

---

To invoke the trace facility, you must ALLOCATE (in MVS) or FILEDEF (in CMS) the ddname FSTRACE, FSTRACE4, or FSTRACE5 to the terminal or to a file with a logical record length (LRECL) of 80 bytes and a record format (RECFM) of F, FB, or V.

To allocate FSTRACE, FSTRACE4, or FSTRACE5 to the terminal, issue the following command within FOCUS:

In TSO:     DYNAM ALLOC FILE {FSTRACE|FSTRACE4|FSTRACE5} DA \*

In CMS:     CMS FILEDEF {FSTRACE|FSTRACE4|FSTRACE5} TERMINAL

To assign the output to a file, issue the following command:

In TSO:     DYNAM FILE {FSTRACE|FSTRACE4|FSTRACE5} DA *datasetname* -  
               SPACE 5,1 CYLINDERS LRECL 80 RECFM FB -  
               BLKSIZE 1600 CATLG UNIT SYSDA

In CMS:     CMS FILEDEF {FSTRACE|FSTRACE4|FSTRACE5}  
               DISK *filename filetype filemode*  
               RECFM V LRECL 80 DISP MOD

For information about JCL for batch access, see Chapter 2, *Getting Started*.

## 9.2.2 Turning Off the Trace Facility

---

To turn off the trace facility, issue the following command:

In TSO:     DYNAM FREE FILE {FSTRACE|FSTRACE4|FSTRACE5}

In CMS:     CMS FILEDEF {FSTRACE|FSTRACE4|FSTRACE5} CLEAR

## 9.3 Verifying the MVS Environment

---

To verify your batch environment, please check the following:

- The Software AG load library is in STEPLIB.
- The ddname DDCARD is allocated to the ADARUN parm card with the correct SVC number.
- The IBI ADABAS.LOAD library is allocated to either STEPLIB, FOCLIB, or USERLIB.

To verify your online environment, please check the following:

- The SOFTWARE.AG.LOAD library is allocated to STEPLIB in either your TSO logon procedure or your LINKLIST (check with your systems programmer).
- The IBI ADABAS.LOAD library is allocated to FOCLIB or USERLIB.

If an abend occurs under MSO on every report request, verify that GENFADL was run to identify the ADABAS SVC number to FOCUS. The GENFADL is run to create FADALINK in FOCLIB.LOAD. The SVC number is the one contained in ADALNK. Please check the offset in FADALINK at hex '84' to see what SVC number is being used.

## 9.4 Verifying the CMS Environment

---

To verify the CMS environment, use FILELIST to confirm access to the following:

- ADAUSER TEXT.
- NUCXTNTS EXEC.
- FOCADA ERRORS.

To run AUTOADBS, use FILELIST to confirm access to the following:

- PREDDB\* FOCADBS.
- PREDEL\* FOCADBS.
- PREDDB\* MASTER.
- PREDEL\* MASTER.

If you do not have access to any of these files, refer to the installation procedure for the CMS environment in Appendix A, *Installation Instructions*.

## 9.5 Verification of Installation and AUTOADBS

---

To verify that you have correctly installed the ADABAS Interface and the AUTOADBS facility, execute the following report request:

```
TABLE FILE PREddb
PRINT FILE_ELEMENT
IF RECORDLIMIT EQ 10
END
```

If the request fails, depending upon the error message, the ADABAS Interface or AUTOADBS was not properly installed. In case of a failure, review the installation procedure in Appendix A, *Installation Instructions*.

## 9.6 Follow-up

---

If the cause of an error is still not apparent, you must:

- Reduce the FOCEXEC to its bare minimum. Remove all possible JOINS, DEFINES, DBAs, and fields from report requests and/or Master and Access Files.
- Supply IBI with the following information:
  - Master File.
  - Access File.
  - FOCEXEC.
  - FSTRACE.
  - FSTRACE4.
  - FSTRACE5, when applicable. IBI will request this trace when necessary.
  - ADABAS ADAREP report.
  - Screen prints of all error messages produced during this troubleshooting session.
  - CLIST or batch JCL.

# A Installation Instructions

---

---

## Topics:

- Pre-installation and Maintenance Requirements
- Installing the ADABAS Interface in MVS
- Installing the ADABAS Interface in CMS

This appendix describes how to install the ADABAS Interface in the MVS and VM/CMS operating environments. These instructions apply to FOCUS Release 7.0 and higher.

Installation prerequisites are explained in Section A.1, *Pre-installation and Maintenance Requirements*. Installation instructions are explained in Section A.2, *Installing the ADABAS Interface in MVS*, and Section A.3, *Installing the ADABAS Interface in CMS*.

## A.1 Pre-installation and Maintenance Requirements

---

Before you install the ADABAS Interface, you should be aware of installation prerequisites and consider maintenance procedures that affect the installation process. This section describes pre-installation and maintenance requirements. They apply to both MVS and CMS, except where noted.

The instructions in this appendix assume that the person performing the installation and maintenance procedures has a working knowledge of MVS or CMS. If you are installing the ADABAS Interface with the FOCUS Multi-Session Option (MSO), MSO knowledge is required. Knowledge of FOCUS and ADABAS is not required.

Your ADABAS database administrator should provide you with site-specific information.

Read this appendix thoroughly before installing the ADABAS Interface to ensure correct installation.

## A.1.1 Software Requirements

---

Before you install the ADABAS Interface, review the following list of software requirements:

- ADABAS must be installed and fully operational. If it is not, contact your ADABAS database administrator.
- FOCUS must be installed and fully operational. If it is not, contact your FOCUS database administrator or consult the appropriate FOCUS installation guide for installation instructions.

You also need to know the FOCUS release. There are two ways to identify your release:

- Check the label of the distribution tape used to install FOCUS; the release number is printed on it.
- Start FOCUS if it is already installed. The release is displayed each time you begin a session.

Every copy of FOCUS has a release number.

## A.1.2 Memory Requirements

---

Enough main memory must be available to execute FOCUS, ADABAS, and the ADABAS Interface. The memory needed is approximately 1440K for FOCUS and 300K for the Interface. Refer to the appropriate ADABAS documentation for ADABAS requirements.

The Interface is fully reentrant and can run in the 31-bit operating environment.

## A.1.3 Maintenance

---

There are no maintenance procedures that you must perform regularly to ensure proper functioning of the ADABAS Interface. However, the following two situations require maintenance:

- If you install a new release of FOCUS, you must also reinstall the ADABAS Interface from the same release.
- If you receive a program temporary fix (PTF) that affects the ADABAS Interface, it will be accompanied by a cover letter containing installation instructions. If you still have installation questions after reading the cover letter, contact Information Builders Customer Support Services (CSS) in New York at (800) 736-6130 or (212) 736-6130, or your local Information Builders representative.

## A.1.4 Using the DDCARD ddname to Specify Site Defaults

---

ADABAS supports the use of the DDCARD ddname to specify site or session defaults for ADABAS, including the default database number, mode (multiple vs. single user), and other parameters.

If you choose a Supervisor Call (SVC) number or database number other than the default, you must allocate the DDCARD ddname to the ADARUN parameter file in your FOCUS CLIST or batch JCL for MVS. Your ADABAS database administrator can provide the name of the ADARUN parameter file for your site.

In this appendix, examples refer to the ADARUN parameter file as follows:

```
SOFTWARE . AG . ADARUN
```

In MVS/TSO, batch, and CMS, the ADABAS Interface recognizes this DDCARD ddname and, if allocated, respects the specified defaults.

**Note:** If the database number is specified with the DBNO attribute in an Access File, the Interface uses the specified number for the ADABAS call. If an Access File does not contain a value for the DBNO attribute, the Interface uses the default specified in the DDCARD ddname. If the DDCARD ddname is not allocated, the Interface uses the default DBNO of 0. Therefore, the order of the DBNO selection is:

1. Access File
2. DDCARD ddname
3. Default of DBNO=0

For more information on the DDCARD ddname, consult your Software AG documentation.

**Note:** MSO does not support the use of the DDCARD ddname. The SVC number is contained in the ADABAS load library module ADALNK. During installation of the ADABAS Interface, the SVC number is placed in the FADALINK load module, from which it is read. See the section on executing the ADABAS Interface under MSO in Chapter 2, *Getting Started*, for more information.

## A.2 Installing the ADABAS Interface in MVS

---

This section describes how to install the ADABAS Interface in the MVS/TSO, MVS batch, and FOCUS Multi-Session Option (MSO) environments. The Interface is distributed in ready-to-execute form.

The installation process consists of the following basic tasks:

1. Unload the distribution tape.
2. Allocate the ADABAS Interface load modules.
3. Access the ADABAS load library supplied by Software AG.
4. Install the ADABAS Interface in the MSO environment (optional).
5. Prepare the run-time libraries.
6. Install the AUTOADBS facility.
7. Test the ADABAS Interface installation.

**Note:** The discussions in this section assume that all of your site's FOCUS and ADABAS Interface libraries are cataloged under the same MVS high-level qualifier.

### A.2.1 Unloading the Distribution Tape

---

The ADABAS Interface is distributed on the same tape/cartridge as the standard FOCUS product. Following are the attributes for the partitioned datasets (PDSs), or libraries, needed to use the Interface:

Dataset	Allocation Parameters
ADABAS.LOAD	SPACE=(CYL,(1,1,5)), DCB=(LRECL=0,BLKSIZE=13030,RECFM=U)
ADABAS.DATA	SPACE=(CYL,(1,1,5)), DCB=(LRECL=80,BLKSIZE=1600,RECFM=FB)

Both of these PDSs are in IEBCOPY dump format:

- The ADABAS.LOAD library contains the ADABAS Interface, distributed as a fully executable load module named ADBSINX.
- The ADABAS.DATA library contains ADABAS Interface error messages and the files required by AUTOADBS.

To unload the distribution tape, perform these steps:

1. Allocate the required disk space for each PDS.
2. Create an IEBCOPY procedure like this sample JCL and submit it for execution

```
//UNLOAD EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//IN15 DD DSN=ADABAS.LOAD,DISP=OLD,
// UNIT=tape,LABEL=(15,SL,EXPDT=98000),VOL=SER=volser
//IN16 DD DSN=ADABAS.DATA,DISP=OLD,
// UNIT=tape,LABEL=(16,SL,EXPDT=98000),VOL=SER=volser
//OUT15 DD DSN=highlvl.ADABAS.LOAD,DISP=OLD
//OUT16 DD DSN=highlvl.ADABAS.DATA,DISP=OLD
//SYSIN DD *
COPY INDD=IN15,OUTDD=OUT15
COPY INDD=IN16,OUTDD=OUT16
/*
```

where:

*tape* Is the distribution tape or cartridge format.

*volser* Is the tape serial number.

*highlvl* Is the high-level qualifier for the ADABAS libraries.

Upon successful completion, two new ADABAS Interface libraries are unloaded from the tape and cataloged on your system.

## A.2.2 Allocating the Interface Load Modules

---

The ADABAS Interface consists of the member ADBSINX in the ADABAS.LOAD library.

At execution time, FOCUS searches for the ADABAS Interface load module in the program library allocated to ddname USERLIB. If USERLIB is not allocated or if it does not contain the ADABAS Interface module, FOCUS searches in ddname STEPLIB. This search order gives the site three options:

- In the STEPLIB allocation, concatenate ADABAS.LOAD to the FOCUS library FOCLIB.LOAD. For example, in your JCL for batch access, specify the following:

```
//STEPLIB DD DSN=highlvl.FOCLIB.LOAD,DISP=SHR
//          DD DSN=highlvl.ADABAS.LOAD,DISP=SHR
//          DD DSN=SOFTWARE.AG.LOAD,DISP=SHR
```

- Copy the ADABAS Interface module ADBSINX from the ADABAS.LOAD library with the NONUM option into the FOCLIB.LOAD library. Then allocate the FOCLIB.LOAD library to ddname STEPLIB. For example, in your JCL for batch access, specify the following:

```
//STEPLIB DD DSN=highlvl.FOCLIB.LOAD,DISP=SHR
//          DD DSN=SOFTWARE.AG.LOAD,DISP=SHR
```

- Allocate the ADABAS.LOAD library as ddname USERLIB. For example, in your JCL for batch access, specify the following:

```
//STEPLIB DD DSN=highlvl.FOCLIB.LOAD,DISP=SHR
//          DD DSN=SOFTWARE.AG.LOAD,DISP=SHR
//USERLIB DD DSN=highlvl.ADABAS.LOAD,DISP=SHR
```

All three options are roughly equivalent in overhead at execution time; the third option, however, requires a small amount of additional storage (approximately 26K) for buffers.

For interactive MVS/TSO, apply the same options. However, you cannot allocate the ddname STEPLIB dynamically using the ALLOCATE command. If you use the STEPLIB ddname online, you must allocate it in the TSO logon procedure or in your LINKLIST. The ddname USERLIB may be allocated dynamically, even within FOCUS, any time prior to the first call to the ADABAS Interface.

## A.2.3 Accessing the Software AG Load Library

---

You need access to the Software AG load library in order for the ADABAS Interface to communicate with the ADABAS DBMS. This library, SOFTWARE.AG.LOAD, contains the ADABAS load modules and the ADABAS file named ADAUSER.

The SOFTWARE.AG.LOAD library must be included in the STEPLIB allocation or in its functional equivalent. Contact your ADABAS database administrator if you are unsure of the name of this load library at your site.

For batch mode, concatenate the SOFTWARE.AG.LOAD library with the FOCLIB.LOAD and ADABAS.LOAD libraries, or with the FOCLIB.LOAD library, in the STEPLIB allocation. See the examples in Section A.2.2, *Allocating the Interface Load Modules*.

For interactive MVS/TSO access, select the option appropriate for your site to access SOFTWARE.AG.LOAD:

- Include the load library in the STEPLIB allocation in the TSO logon procedure.
- Include the load library in the STEPLIB allocation in the LINKLIST.
- If you are calling FOCUS from ISPF, use the ddname ISPLLIB, instead of STEPLIB, for the load library. The ddname ISPLLIB can be set dynamically and must be allocated before entering ISPF (for example, in the CLIST that calls ISPF). The ddname ISPLLIB cannot be reassigned using the LIBDEF command.

## A.2.4 Preparing the Run-time Libraries

---

At this point, you may need to create or modify five run-time libraries. Following sections describe how to allocate these libraries with a CLIST or batch JCL. A sample CLIST and sample batch JCL are shown in Section A.2.7, *Testing the ADABAS Interface Installation*.

Each Interface user must have access to the five libraries, which are described below.

### The MASTER Library

---

The MASTER library is created when FOCUS is installed. The library is allocated to ddname MASTER, and its members are FOCUS Master Files.

FOCUS uses a Master File to describe the data in an ADABAS file. Each ADABAS file accessed by the Interface must be described to FOCUS as a Master File or as a segment in a Master File.

## The FOCADBS (or ACCESS) Library

---

Using your site criteria, allocate a PDS for your Access Files as ddname FOCADBS or ACCESS. Two suggested names for this library are *userid.FOCADBS.DATA* or *userid.ACCESS.DATA*.

**Note:** In this appendix, we use the term FOCADBS library to refer to either the FOCADBS or ACCESS library, depending on the name chosen at your site.

Access Files correspond to Master Files and provide additional ADABAS file characteristics such as the ADABAS database number, file number, and access type.

You create and allocate your own Access File library. However, some sites create one centrally stored PDS for all Access Files. In that case, you can concatenate your individual Access Files with the site's Access Files.

The DCB requirements of the FOCADBS library are the same as those for the FOCEXEC and MASTER libraries. It must be a PDS with LRECL=80, RECFM=F or FB, and a BLKSIZE that is any multiple of 80. The amount of required space depends on the size and number of Access Files you use. Following are recommended space allocation and DCB values:

Dataset	Allocation Parameters
<i>userid.FOCADBS.DATA</i> or <i>userid.ACCESS.DATA</i>	SPACE=(TRK,(5,1,20)), DCB=(LRECL=80,BLKSIZE=1760,RECFM=FB)

For complete information about Master and Access Files, see Chapter 5, *Describing ADABAS Files to FOCUS*.

## The FOCEXEC Library

---

The FOCEXEC library is created when FOCUS is installed. The library is allocated to ddname FOCEXEC, and its members are FOCUS procedures (FOCEXECs).

## The ADABAS Interface Load Library

---

The ADABAS.LOAD library is supplied on the distribution tape. This library contains ADABAS Interface software and does not require additional preparation.

## The ERRORS Library

---

The ERRORS library is created when FOCUS is installed. Like FOCUS, the ADABAS Interface has its own set of error messages, which normally include an ADABAS response code in case of failure. The *highlvl.ADABAS.DATA* library contains the ADABAS Interface error messages and must be allocated to ddname ERRORS.

The following JCL concatenates the ADABAS.DATA PDS in the ERRORS allocation to properly generate ADABAS Interface error messages:

```
//ERRORS DD DSN=highlvl.ERRORS.DATA,DISP=SHR
//          DD DSN=highlvl.ADABAS.DATA,DISP=SHR
```

Failure to allocate the ADABAS.DATA library properly displays the following message instead of an appropriate ADABAS Interface error message:

```
FOC000 ERROR MESSAGE TEXT MISSING
```

The number that follows FOC is the error message number. If this message displays, correct your ERRORS allocation to include *highlvl.ADABAS.DATA*.

For individual ADABAS Interface error messages, consult Appendix B, *ADABAS Interface Error Messages*. For details on the ADABAS response codes that may accompany the ADABAS Interface error messages, see your Software AG publication *General Reference Set: Messages and Codes Manual*.

## A.2.5 Installing in the MSO Environment (Optional)

---

If you use the ADABAS Interface in an MSO environment, additional steps are required:

1. Run GENFADL JCL.

Use the sample JCL provided in ADABAS.DATA(GENFADL) to create a new module (FADALINK) from the Software AG module ADALNK. Copy the FADALINK module into the ADABAS.LOAD library. The FADALINK module is required to identify the ADABAS SVC number to FOCUS for communication purposes.

2. Include the necessary libraries in the MSO startup JCL.

- In addition to the FOCLIB.LOAD library in your MSO startup JCL, include the Software AG load library in the DD statement for the STEPLIB allocation.
- Allocate the ADABAS Interface load library, ADABAS.LOAD, to ddname STEPLIB or USERLIB.
- Include the ADABAS.DATA library in the ERRORS allocation.

For example:

```
//STEPLIB DD DSN=highlvl.FOCLIB.LOAD,DISP=SHR
//          DD DSN=SOFTWARE.AG.LOAD,DISP=SHR
//          DD DSN=highlvl.ADABAS.LOAD,DISP=SHR
//ERRORS  DD DSN=highlvl.ERRORS.DATA,DISP=SHR
//          DD DSN=highlvl.ADABAS.DATA,DISP=SHR
```

Optionally, include allocations for the MASTER and FOCADBS libraries in the MSO startup JCL to allow access to centrally stored Master and Access Files.

## A.2.6 Installing AUTOADBS

---

The ADABAS Interface is distributed with a full-screen, automated procedure named AUTOADBS, which simplifies the process of describing ADABAS files to FOCUS. AUTOADBS generates Master and Access Files for specified ADABAS files by using the file information cataloged in the Predict dictionary. For instructions on how to use AUTOADBS, see Chapter 4, *The AUTOADBS Facility*.

The AUTOADBS installation process consists of the following tasks:

1. Edit the GENFADLA member in the ADABAS.DATA library and submit the JCL.
2. Determine which Predict dictionaries require access and create menu selection choices.
3. Edit the PREddb and PREDEL members in the Access File PDS.
4. Determine the value for the CALLTYPE attribute.
5. Decide whether the SEQFIELD attribute is optional or required.
6. Change the initial dataset defaults.

## Editing the GENFADLA Member

---

Edit the JCL found in *highlvl*.ADABAS.DATA(GENFADLA), using these steps:

1. Add a job card to the JCL.
2. Change *highlvl* to the high-level qualifier used for ADABAS.DATA, and *userid* to the high-level qualifier used for FOCEXEC.DATA, MASTER.DATA, and FOCADBS.DATA.
3. Submit the job to copy and rename the AUTOADBS files from the installation datasets to the run-time libraries.

## Determining Which Predict Dictionaries Require Access

---

If you use only one Predict dictionary as a source for FOCUS file descriptions, skip this section and continue with the next task (editing members in the Access File library).

For AUTOADBS to access several Predict dictionaries concurrently, you must do the following:

1. Determine which Predict dictionaries need to be accessed and create menu selection choices. For each dictionary, select a one-character identifier. This value uniquely identifies the dictionary for reporting purposes and is displayed on the Main Menu of AUTOADBS. Select a letter A through Z, a number 0 through 9, or blank for the identifier.

For example, if you have a test and production Predict dictionary, you might select T and P as the identifiers.

2. For each menu identifier, copy and rename the following files. For each member name, the suffix *n* represents the identifier. (Remember, whenever you see *userid*.FOCADBS.DATA, you can substitute *userid*.ACCESS.DATA if applicable at your site.)

From:	To:
<i>userid</i> .MASTER.DATA(PREDDB)	<i>userid</i> .MASTER.DATA(PREDDB <i>n</i> )
<i>userid</i> .MASTER.DATA(PREDEL)	<i>userid</i> .MASTER.DATA(PREDEL <i>n</i> )
<i>userid</i> .FOCADBS.DATA(PREDDB)	<i>userid</i> .FOCADBS.DATA(PREDDB <i>n</i> )
<i>userid</i> .FOCADBS.DATA(PREDEL)	<i>userid</i> .FOCADBS.DATA(PREDEL <i>n</i> )

3. Edit the AUTOADBS FOCEXEC to include the menu identifier. In the dataset *userid*.FOCEXEC.DATA(AUTOADBS), locate the line that reads

```
-DEFAULT &DICT_FILES=' '
```

and replace the blank value with the identifier.

## Installation Instructions

You can have more than one character. In that case, the first character is used as the default dictionary on the AUTOADBS Main Menu. For example, if you choose P and T as your identifiers and the production dictionary is the default, the line reads:

```
-DEFAULT &DICT_FILES='PT'
```

**Note:** The characters must be entered in uppercase.

4. Repeat Step 2 for each of the identifiers selected in Step 1.

## Editing the PREddb and PREDEL Members in the Access File PDS

---

Edit the PREddb and PREDEL members in *userid.FOCADBS.DATA* (or *userid.ACCESS.DATA*). If you have several Predict dictionaries with associated identifiers, repeat these steps for all of the PREddb and PREDEL members created in the FOCADBS.DATA library described in the section, *Determining Which Predict Dictionaries Require Access*, Section A.2.6, *Installing AUTOADBS*.

1. Update the ADABAS database number and file number for the Predict dictionary in each member. Locate the attributes DBNO=001 and FILENO=012 and change these values to the actual values for your site's Predict dictionary. You must change two occurrences in the member PREddb and *all three* occurrences in the member PREDEL.

```
$ FILENAME=PREddb, SUFFIX=ADBSINX, $
RELEASE=6, OPEN=YES, $
$ ADABAS FILE = SYSDIC-FI
SEGNAME=S02, ACCESS=ADBS, DBNO=001, FILENO=012,
CALLTYPE=RL, SEQFIELD=FILE_NAME_FI,
FIELD=FILE_NAME_FI, TYPE=DSC, $
FIELD=FILE_TYPE, TYPE=DSC, $
FIELD=L_FNR, TYPE=DSC, $
SEGNAME=DI0602, ACCESS=MU, DBNO=001, FILENO=012, $ RIPP_NR
FIELD=RIPP_NR, TYPE=DSC, $
```

**Figure A-1.** Example of a PREddb Access File

```
$ FILENAME=PREDEL, SUFFIX=ADBSINX, $
$ ADABAS FILE = SYSDIC-EL
RELEASE=6, OPEN=YES, $
SEGNAME=S03, ACCESS=ADBS, DBNO=001, FILENO=012,
CALLTYPE=RL, SEQFIELD=FIELD_SEQ, $
FIELD=FIELD_SEQ, TYPE=SPR, $
FIELD=FILE_NAME_EL, TYPE=, NU=NO, $
FIELD=FIELD_SEQUEN, TYPE=DSC, NU=NO, $
SEGNAME=BD0103, ACCESS=MU, DBNO=001, FILENO=012, $ COMMENTS
SEGNAME=BQ0603, ACCESS=MU, DBNO=001, FILENO=012, $ DERIVATION_GROUP
```

**Figure A-2.** Example of a PREDEL Access File

2. If your Predict dictionaries are password-protected, you must insert the password value in each member. Add the PASSWORD attribute after *all three* occurrences of the FILENO attribute in each member.

The PASSWORD attribute with the value ADADBA is included in the following sample PREDEL Access File:

```

$ FILENAME=PREDEL,SUFFIX=ADBSINX,$
$ ADABAS FILE = SYSDIC-EL
RELEASE=6,OPEN=YES,$
SEGNAM=S03,ACCESS=ADBS,DBNO=001,FILENO=012,PASSWORD=ADADBA,
      CALLTYPE=RL,SEQFIELD=FIELD_SEQ,$
FIELD= FIELD_SEQ,TYPE=SPR,$
FIELD=FILE_NAME_EL,TYPE=,NU=NO,$
FIELD=FIELD_SEQUEN,TYPE=DSC,NU=NO,$
SEGNAM=BD0103,ACCESS=MU,DBNO=001,FILENO=012,PASSWORD=ADADBA,$ COMMENTS
SEGNAM=BQ0603,ACCESS=MU,DBNO=001,FILENO=012,PASSWORD=ADADBA,$

```

*Figure A-3. Example of a PREDEL Access File With the PASSWORD Attribute*

## Determining the CALLTYPE Value

---

Determine how AUTOADBS treats the CALLTYPE value you supply. The CALLTYPE attribute appears in all generated Access Files. Its value indicates whether the ADABAS Interface needs to construct Read Logical (RL) or FIND calls for retrieval. You have two choices:

- By default, AUTOADBS assigns RL. Override this default by entering FIND for the CALLTYPE value.
- If you do not want users to change the CALLTYPE to FIND, restrict the acceptable value to RL. Edit the AUTOADBS member in *userid.FOCEXEC.DATA* and locate the line that reads:

```
-DEFAULT &REQ_RL=NO
```

Change NO to YES. If you do this, AUTOADBS will accept *only* the RL value.

## Deciding Whether the SEQFIELD Attribute Is Optional or Required

---

We recommend using the SEQFIELD option. This field (an index) is used when there are no IF or WHERE clauses in the report request. This option avoids an L2 call (read physical) through the entire database. Using the index is more efficient in most cases.

Determine if AUTOADBS requires the SEQFIELD value. The SEQFIELD attribute is specified to use an ADABAS descriptor or superdescriptor's inverted list for retrieval by default. You have two choices:

- By default, AUTOADBS requires a value for the SEQFIELD attribute. If you accept the default, the user must enter a SEQFIELD value for each ADABAS file when running AUTOADBS. The value will be included in the generated Access File.
- You can change the default to make the SEQFIELD value optional. To do so, edit the AUTOADBS member in *userid.FOCEXEC.DATA* and locate the line that reads:

```
-DEFAULT &REQ_SEQFIELD=YES
```

Change YES to NO. If the user chooses to specify a SEQFIELD value when running AUTOADBS, AUTOADBS will include the value in the generated Access File. However, a value is not required.

## Changing the Initial Dataset Defaults

---

AUTOADBS displays default output dataset names for the Master File, Access File, and FOCDEF file. These defaults follow standard IBI naming conventions. The defaults are changed on a per-user basis by replacing the values on the screen and then logging them to disk for subsequent use. Alternatively, the AUTOADBS FOCEXEC can be modified to include site-specific naming conventions for all users.

To customize the initial dataset defaults for all users, edit the following lines in the AUTOADBS member of *userid.FOCEXEC.DATA*. Change the expression on the right-hand side of the equal sign as appropriate. The resulting expression must retain a length of 44 characters (&USERID accounts for eight characters). You only need to customize the following lines

```
-SET &DSNP0=&USERID || ' .FOCADBS .DATA           ' ;  
-SET &DSNM0=&USERID || ' .MASTER .DATA           ' ;  
-SET &DSNF0=&USERID || ' .FOCADBS .DATA           ' ;  
-SET &DSND0=&USERID || ' .FOCDEF .DATA           ' ;
```

where:

&DSNP0	Is the menu parameter log dataset.
&DSNM0	Is the Master File output dataset.
&DSNF0	Is the Access File output dataset.

&amp;DSNDO

Is the FOCDEF TableTalk Help output dataset.

## A.2.7 Testing the ADABAS Interface Installation

To test the ADABAS Interface installation and file descriptions used by AUTOADBS, use the following samples and FOCUS report request. Modify them according to your site's requirements. You may need to change the dataset name for the ADARUN parameter file. (See Section A.1.4, *Using the DDCARD ddname to Specify Site Defaults.*)

### Test Sample CLIST for Interactive MVS/TSO

Specify your site information and execute the following CLIST

```

/*****
/*          *Test CLIST FOR testing FOCUS w/ ADABAS Interface*          */
/*  Note: This CLIST assumes that the Software AG load library has been  */
/*  allocated to ddname STEPLIB in the TSO logon proc or system linklist. */
/*  'highlvl' represents the high-level qualifier chosen for the        */
/*  FOCUS libraries.                                                    */
/*****
/*
/*  Allocate FUSELIB.LOAD and ADABAS.LOAD to ddname USERLIB            */
/*
/*      ALLOC F(USERLIB) DA('highlvl.FUSELIB.LOAD' -                    */
/*                          'highlvl.ADABAS.LOAD') SHR REU              */
/*
/*  Allocate ddnames that will be used in the FOCUS session.          */
/*  Assure that the PDses allocated to FOCEXEC, MASTER, and            */
/*  FOCADBS are the ones that contains the AUTOADBS files copied      */
/*  in the AUTOADBS Installation step.                                  */
/*
/*      ALLOC F(ERRORS)  DA('highlvl.ERRORS.DATA' -                    */
/*                          'highlvl.ADABAS.DATA') SHR REUSE           */
/*      ALLOC F(FOCEXEC) DA('userid.FOCEXEC.DATA') SHR REUSE          */
/*      ALLOC F(DDCARD)  DA('SOFTWARE.AG.ADARUN') SHR REUSE           */
/*      ALLOC F(MASTER)  DA('userid.MASTER.DATA') SHR REUSE          */
/*      ALLOC F(FOCADBS) DA('userid.FOCADBS.DATA') SHR REUSE         */
/*
/*  The actual call to the FOCUS module follows                        */
/*      CALL 'highlvl.FOCLIB.LOAD(FOCUS)'

```

where:

*highlvl* Is the high-level qualifier for your site's FOCUS production libraries.

*userid* Is the high-level qualifier for your private version of a library.

The result is a FOCUS session in interactive MVS/TSO. Once in FOCUS, enter this simple report request:

```

TABLE FILE PREDBB
PRINT FILE_ELEMENT
IF RECORDLIMIT EQ 10
END

```

If the ADABAS Interface is properly installed, the result is a FOCUS report that displays data from the first ten records in your Predict dictionary.

## Installation Instructions

**Note:** If you customize AUTOADBS to run with multiple copies of the Predict dictionary, append the suffix for the identifier to PREDDDB in the request. For example, if you choose P or T as a suffix, specify

```
TABLE FILE PREDDBP
```

or:

```
TABLE FILE PREDDBT
```

If you do not receive a report (0 records), make sure that the Predict dictionary contains data. The Predict dictionary is identified by the DBNO and FILENO in FOCADBS.DATA(PREDDB) and FOCADBS.DATA(PREDEL).

If you receive any ADABAS Interface error messages and/or ADABAS response codes, check your installation against the instructions in this appendix. Also consult Appendix B, *Interface Error Messages*, and the appropriate Software AG documentation, or call Information Builders for assistance.

## Test Sample JCL

Specify your site information and submit the following job

```
/** * * * * JOB CARD Information goes here * * * *  
/**  
/**Note that 'highlvl' represents the high-level qualifier chosen for  
/**the FOCUS libraries. For the ddnames MASTER, FOCADBS, and FOCEXEC,  
/**make certain that you are pointing to the libraries that contain  
/**the files used by AUTOADBS as described above.  
/**  
//STEP01 EXEC PGM=FOCUS  
//STEPLIB DD DSN=SOFTWARE.AG.LOAD,DISP=SHR *The ADABAS load library*  
// DD DSN=highlvl.FOCLIB.LOAD,DISP=SHR  
// DD DSN=highlvl.ADABAS.LOAD,DISP=SHR  
//USERLIB DD DSN=highlvl.FUSELIB.LOAD,DISP=SHR  
//ERRORS DD DSN=highlvl.ERRORS.DATA,DISP=SHR  
// DD DSN=highlvl.ADABAS.DATA,DISP=SHR  
//FOCEXEC DD DSN=userid.FOCEXEC.DATA,DISP=SHR  
//MASTER DD DSN=userid.MASTER.DATA,DISP=SHR  
//FOCADBS DD DSN=userid.FOCADBS.DATA,DISP=SHR  
//DDCARD DD DSN=SOFTWARE.AG.ADARUN,DISP=SHR  
//SYSOUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *  
TABLE FILE PREDDB  
PRINT FILE_ELEMENT  
IF RECORDLIMIT EQ 10  
END  
FIN  
/**
```

where:

*highlvl* Is the high-level qualifier for your site's FOCUS production libraries.

*userid* Is the high-level qualifier for your private version of a library.

If the ADABAS Interface is properly installed, the result is a FOCUS report that displays data from the first ten records in your Predict dictionary.

**Note:** If you customize AUTOADBS to run with multiple copies of the Predict dictionary, append the suffix for the identifier to PREDDDB in the request. For example, if you choose P or T as a suffix, specify

```
TABLE FILE PREDDBP
```

or:

```
TABLE FILE PREDDBT
```

If you do not receive a report (0 records), make sure that the Predict dictionary contains data. The Predict dictionary is identified by the DBNO and FILENO in FOCADBS.DATA(PREDDDB) and FOCADBS.DATA(PREDEL).

If you receive any ADABAS Interface error messages and/or ADABAS response codes, check your installation against the instructions in this appendix. Also consult Appendix B, *Interface Error Messages*, and the appropriate Software AG documentation, or call Information Builders for assistance.

## A.3 Installing the ADABAS Interface in CMS

---

This section describes how to install the ADABAS Interface in the CMS environment. The ADABAS Interface is distributed in ready-to-execute form. The installation requires only a tape-to-disk load and a copy step to move the components to the appropriate libraries.

The installation process consists of the following basic tasks:

1. Prepare the ADABAS Interface load libraries.
2. Create the ADAUSER TEXT file.
3. Access the ADABAS nucleus.
4. Install AUTOADBS.
5. Test the ADABAS Interface installation.

## A.3.1 Preparing the ADABAS Interface Load Libraries

---

The ADABAS Interface is distributed on the same tape/cartridge as the standard FOCUS product. The necessary ADABAS Interface load libraries, ADABAS Interface error message files, and AUTOADBS files are copied (during the COPYFS step) to the FOCUS production disk when FOCUS for CMS is installed.

If you have not completed the standard FOCUS installation, do it now before continuing.

FOCUS must be installed and fully operational. If it is not, contact your FOCUS database administrator or consult the appropriate FOCUS installation guide for instructions.

## A.3.2 Creating the ADAUSER TEXT File

---

To establish communication with the ADABAS nucleus, standard FOCUS requires an ADABAS file named ADAUSER TEXT. To create ADAUSER TEXT, you need access to the following files and libraries:

- The Software AG source file ADAUSER ASSEMBLE. This file usually resides on the disk of the ADABAS database administrator.
- Macro libraries (for example, DMSGPI MACLIB, HCPGPI MACLIB, OSPSI MACLIB). If you are unsure which macro libraries are required at your site, ask your CMS systems staff.
- The Software AG macro library. The naming convention is usually

ADAVxxx MACLIB

where:

xxx                      Is the ADABAS version number (for example, ADAV524 MACLIB for ADABAS version 5.2.4).

Perform these steps to create the ADAUSER TEXT file and copy it to the FOCUS production disk:

1. From the CMS Ready prompt, specify the search order of the macro libraries with the CMS GLOBAL command:

```
GLOBAL MACLIB DMSGPI ADAVxxx HCPGPI OSPSI
```

2. Assemble the ADAUSER TEXT file:

```
ASSEMBLE ADAUSER
```

This command creates the ADAUSER TEXT file on your A disk. Consult your CMS systems staff if any errors occur.

3. Copy the ADAUSER TEXT file to the FOCUS production disk. If a file already exists, replace it.

### A.3.3 Accessing the ADABAS Nucleus

---

To establish a link between your CMS FOCUS machine and the ADABAS nucleus, execute NUCXTNTS EXEC. This EXEC usually resides on the disk of the ADABAS database administrator. Supplied by Software AG, it provides ADABAS access from your programs in CMS. From the CMS Ready prompt, enter:

```
EX NUCXTNTS
```

You must execute the EXEC once per CMS session. Include this command in a PROFILE or in the EXEC that is used to call FOCUS. For global access, copy the NUCXTNTS EXEC to the FOCUS production minidisk.

### A.3.4 Installing AUTOADBS for CMS

---

The ADABAS Interface is distributed with an automated procedure that simplifies the process of describing ADABAS files to FOCUS. It is a full-screen procedure named AUTOADBS. AUTOADBS generates MASTER and Access Files for specified ADABAS files by using the file information stored in the Predict dictionary. For instructions on how to use AUTOADBS, see Chapter 4, *The AUTOADBS Facility*.

The files used by AUTOADBS are automatically copied to the FOCUS production disk; however, you may need to edit these files to conform to your site requirements.

The AUTOADBS installation process consists of the following tasks:

1. Determine which Predict dictionaries require access and create menu selection choices.
2. Edit the Access Files PREDDB FOCADBS and PREDEL FOCADBS.

**Note:** You can assign the filetype ACCESS to your Access Files. In this appendix, we use the term FOCADBS to refer to either FOCADBS or ACCESS, depending on the name chosen at your site.

3. Determine the value for the CALLTYPE attribute.
4. Decide whether the SEQFIELD attribute is optional or required.

## Determining Which Predict Dictionaries Require Access

---

If you have only one Predict dictionary to use as a source for FOCUS file descriptions, skip this section and continue with the next task of editing the database number (DBNO) and file number (FILENO).

In order for AUTOADBS to access several Predict dictionaries concurrently, you must do the following:

1. Determine which Predict dictionaries need to be accessed and create menu selection choices. For each dictionary, select a one-character identifier. This value uniquely identifies the dictionary for reporting purposes and is displayed on the Main Menu of AUTOADBS. Select a letter A through Z, a number 0 through 9, or blank for the identifier.

For example, if you have a test and production Predict dictionary, you might select T and P as the identifiers.

2. For each menu identifier, copy and rename the following files. For each filename, the suffix *n* represents the identifier.

<b>From:</b>	<b>To:</b>
PREddb MASTER	PREddb <sub>n</sub> MASTER
PREDEL MASTER	PREDEL <sub>n</sub> MASTER
PREddb FOCADBS	PREddb <sub>n</sub> FOCADBS
PREDEL FOCADBS	PREDEL <sub>n</sub> FOCADBS

3. Edit AUTOADBS FOCEXEC to include all the menu identifiers. Locate the line that reads  
`-DEFAULT &DICT_FILES=' '`  
and replace the blank value with the identifier.

You can have more than one character. In that case, the first character is used as the default dictionary on the AUTOADBS Main Menu. For example, if you choose P and T as your identifiers and the production dictionary is the default, the line reads:

```
-DEFAULT &DICT_FILES='PT'
```

**Note:** The characters must be entered in uppercase.

## Editing the PREddb and PREDEL Access Files

Edit the Access Files named PREddb FOCADBS and PREDEL FOCADBS. If you have several Predict dictionaries with associated identifiers, repeat these steps for all of the PREddb and PREDEL Access Files created in the section, *Determining Which Predict Dictionaries Require Access*, Section A.3.4, *Installing AUTOADBS for CMS*.

1. Update the ADABAS database number and file number for the Predict dictionary in each Access File. Locate the attributes DBNO=001 and FILENO=012 and change these values to the actual values for your site's Predict dictionary. You must change one occurrence in the file PREddb and *all three* occurrences in the PREDEL.

```

$ FILENAME=PREddb , SUFFIX=ADBSINX , $
RELEASE=6 , OPEN=YES , $
$ ADABAS FILE = SYSDIC-FI
SEGNAM=S02 , ACCESS=ADBS , DBNO=001 , FILENO=012 ,
      CALLTYPE=RL , SEQFIELD=FILE_REC ,
      KEYFLD=FILE_ELEMENT ,
      IXFLD=FILE_REC
FIELD=FILE_REC , TYPE=NOP , $

```

**Figure A-4. Example of a PREddb Access File**

```

$ FILENAME=PREDEL , SUFFIX=ADBSINX , $
$ ADABAS FILE = SYSDIC-EL
RELEASE=6 , OPEN=YES , $
SEGNAM=S03 , ACCESS=ADBS , DBNO=001 , FILENO=012 ,
      CALLTYPE=RL , SEQFIELD=FIELD_SEQ , $
FIELD= FIELD_SEQ , TYPE=SPR , $
FIELD=FILE_NAME_EL , TYPE= , NU=NO , $
FIELD=FIELD_SEQUEN , TYPE=DSC , NU=NO , $
SEGNAM=BD0103 , ACCESS=MU , DBNO=001 , FILENO=012 , $ COMMENTS
SEGNAM=BQ0603 , ACCESS=MU , DBNO=001 , FILENO=012 , $ DERIVATION_GROUP

```

**Figure A-5. Example of a PREDEL Access File**

## Installation Instructions

2. If your Predict dictionaries are password-protected, you must insert the password value in each member. Add the PASSWORD attribute after *all three* occurrences of the FILENO attribute in each member.

The PASSWORD attribute with the value ADADBA is included in the following sample PREDEL Access File:

```
$ FILENAME=PREDEL,SUFFIX=ADBSINX,$
$ ADABAS FILE = SYSDIC-EL
RELEASE=6,OPEN=YES,$
SEGNAM=S03,ACCESS=ADBS,DBNO=001,FILENO=012,PASSWORD=ADADBA,
CALLTYPE=RL,SEQFIELD=FIELD_SEQ,$
FIELD= FIELD_SEQ,TYPE=SPR,$
FIELD=FILE_NAME_EL,TYPE=,NU=NO,$
FIELD=FIELD_SEQUEN,TYPE=DSC,NU=NO,$
SEGNAM=BD0103,ACCESS=MU,DBNO=001,FILENO=012,PASSWORD=ADADBA,$ COMMENTS
SEGNAM=BQ0603,ACCESS=MU,DBNO=001,FILENO=012,PASSWORD=ADADBA,$
```

*Figure A-6. Example of a PREDEL Access File With the PASSWORD Attribute*

## Determining the CALLTYPE Value

---

Determine how AUTOADBS treats the CALLTYPE value you supply. The CALLTYPE attribute appears in all generated Access Files. Its value indicates whether the ADABAS Interface needs to construct Read Logical (RL) or FIND calls for retrieval. You have two choices:

- By default, AUTOADBS assigns RL. Override this default by entering FIND for the CALLTYPE value.
- If you do not want users to change the CALLTYPE value to FIND, restrict the acceptable value to RL. Edit the AUTOADBS FOCEXEC and locate the line that reads:

```
-DEFAULT &REQ_RL=NO
```

Change NO to YES. AUTOADBS will then accept *only* the RL value.

## Deciding Whether the SEQFIELD Attribute Is Optional or Required

---

We recommend using the SEQFIELD option. This field (an index) is used when there are no IF or WHERE clauses in the report request. This option avoids an L2 call (Read Physical) through the entire database. Using the index is more efficient in most cases.

Determine if AUTOADBS requires the SEQFIELD value. The SEQFIELD attribute is specified to use an ADABAS descriptor or superdescriptor's inverted list for retrieval by default. You have two choices:

- By default, AUTOADBS requires a value for the SEQFIELD attribute. If you accept the default, the user must enter a SEQFIELD value for each ADABAS file when running AUTOADBS. The value will be included in the generated Access File.
- You can change the default to make the SEQFIELD value optional. To do so, edit the AUTOADBS FOCEXEC and locate the line that reads:

```
-DEFAULT &REQ_SEQFIELD=YES
```

Change YES to NO. If the user chooses to specify a SEQFIELD value when running AUTOADBS, AUTOADBS will include the value in the generated Access File. However, a value is not required.

### A.3.5 Testing the ADABAS Interface Installation

---

After you copy the appropriate ADABAS Interface load libraries and ADAUSER TEXT to the FOCUS minidisk and execute the NUCXTNTS EXEC from CMS, test the ADABAS Interface installation:

1. Enter the FOCUS session by typing

```
FOCUS
```

at the CMS Ready prompt.

2. Enter the following simple report request:

```
TABLE FILE PREddb
PRINT FILE_ELEMENT
IF RECORDLIMIT EQ 10
END
```

If the ADABAS Interface is properly installed, the result is a FOCUS report that displays data from the first ten records in your Predict dictionary.

## Installation Instructions

**Note:** If you customize AUTOADBS to run with multiple copies of the Predict dictionary, append the suffix for the identifier to PREDDDB in the request. For example, if you choose P or T as a suffix, specify

```
TABLE FILE PREDDBP
```

or:

```
TABLE FILE PREDDBT
```

If you did not receive a report (0 records), make sure that the Predict dictionary contains data. The Predict dictionary is identified by the DBNO and FILENO in PREDDDB FOCADBS and PREDEL FOCADBS.

If you receive any ADABAS Interface error messages and/or ADABAS response codes, check your installation against the instructions in this appendix. Also consult Appendix B, *Interface Error Messages*, and the appropriate Software AG documentation, or call Information Builders for assistance.

## B Interface Error Messages

---

**Topic:**

- Messages for MVS and CMS

This appendix contains the list of ADABAS Interface error messages.

### B.1 Messages for MVS and CMS

---

Interface error messages are stored in *highlvl*.ADABAS.DATA(FOCADA) for MVS, and in FOCADA ERRORS for CMS.

Issuing a ? n query (where n is the message number) at the FOCUS command level displays a detailed explanation of the message. For example, if you receive the message

```
(FOC4453) ACCESS OF ENTRY SEGMENT NOT ADBS %1%2%3%4
```

you can enter

```
? 4453
```

to get this explanation:

```
(FOC4453) ACCESS OF ENTRY SEGMENT NOT ADBS %1%2%3%4
```

```
The Access File record for the entry segment does not specify ACCESS=ADBS.  
The access mode for the top segment cannot be 'PE' or 'MU'.
```

A complete list of the Interface error and warning messages and their detailed explanations follows. Digits preceded by a percent sign (for example, %1) are placeholders for context-sensitive items, such as filenames and fieldnames. FOCUS replaces each placeholder with the name of the relevant object when it displays the message.

```
(FOC4451) ACCESS FILE NOT FOUND.%1%2%3%4
```

On TSO, the Access File is a member of a partitioned dataset allocated with ddname 'FOCADBS'. The member name sought must correspond to the FOCUS name of the ADABAS database (for example, if the Master File PDS has the name 'CAR', then the Access File must have a member name of 'CAR').

PROBABLE ERROR: Ddname 'FOCADBS' is not allocated. If it is allocated, the DSN parameter may be pointing to the wrong PDS, the member name may be misspelled in the Master File, or the member itself is missing.

## Interface Error Messages

On CMS, the Access File must have filetype 'FOCADBS' and can be located on any accessed disk. The filename of the Access File must correspond to the FOCUS name of the ADABAS database (for example, if the Master File contains 'FILE=EMP', the Access File must be named 'EMPFOCADBS xx' where xx is any accessed disk.)

PROBABLE ERROR: The filename is misspelled or the file is located on an unaccessed disk.

(FOC4452) I/O ERROR READING ACCESS FILE 1%2%3%4

On TSO, the error is probably caused by allocating DDNAME 'FOCADBS' to a sequential dataset rather than a partitioned dataset (PDS).

On CMS, the error represents a genuine I/O error.

In both environments, the Access File format is fixed blocked with an 80 character record length. Access File records are not line numbered.

(FOC4453) ACCESS OF ENTRY SEGMENT NOT ADBS %1%2%3%4

The Access File record for the entry segment does not specify ACCESS=ADBS. The access mode for the top segment cannot be 'PE' or 'MU'.

(FOC4454) NO COUNT FIELD ON OCCURS SEGMENT %1%2%3%4

The segment specified is a periodic group (PE) or multi-value field (MU) as specified in the Access File. The Master File contains the OCCURS parameter, but the value specified is a numeric constant. The value for the OCCURS parameter for ADABAS PE and MU segments must be a field defined in the parent segment with USAGE=I4 and ACTUAL=I1.

(FOC4455) ADBS SEGMENT HAS OCCURS %1%2%3%4

(FOC4456) KEYFLD AND IXFLD INCOMPATIBLE %1%2%3%4

In the Master File, the KEYFLD on the parent segment has a different USAGE type from the ACTUAL format type of the corresponding index field on the descendent segment. They must have comparable format types.

(FOC4457) USAGE AND ACTUAL FORMATS INCOMPATIBLE %1%2%3%4

In the Master File the USAGE format specified is not compatible with the ACTUAL format shown there.

(FOC4458) KEYFLD IS BLANK %1%2%3%4

In the Access File, the access type specified for a segment is ADBS, but no KEYFLD value is specified.

(FOC4459) DESCRIPTOR MUST HAVE INDEX=I %1%2%3%4

The value indicated is specified for an IXFLD parameter in the Access File. It is a valid Master File field, but it is not specified with the INDEX=I parameter. The field must be an ADABAS descriptor, subdescriptor, or superdescriptor, and must be defined with INDEX=I.

(FOC4460) ACCESS KEYWORD NOT VALID %1%2%3%4

The value specified as the ACCESS method for the indicated segment is not ADBS, PE, or MU. The value is missing or misspelled.

- (FOC4461) **IXFLD IS NOT IN OWN SEGMENT %1%2%3%4**  
 The value indicated is specified for an IXFLD parameter in the Access File, but this field is not located in that segment as specified in the Master File. Either the IXFLD value is not correct in the Access File or the field was erroneously placed in another segment.
- (FOC4462) **KEYFLD IS NOT A FOCUS FIELDNAME %1%2%3%4**  
 The value specified in the Access File for the KEYFLD of a segment was the fieldname indicated in the message, but the field is not a fieldname in the Master File. Be sure not to use the ALIAS name as the value for the KEYFLD.
- (FOC4464) **KEYFIELD NOT IN PARENT SEGMENT %1%2%3%4**  
 The value indicated is specified as the KEYFLD value in the Access File for the SEGNAM indicated, but this field is not located in that segment in the Master File. Either the IXFLD value is incorrect in the Access File or the segment specified as the PARENT is incorrect in the Master File.
- (FOC4465) **INVALID ADABAS FILE NUMBER %1%2%3%4**  
 The value specified for the file number on the segment specified is incorrect. It must be a numeric value from 1 - 256.
- (FOC4466) **SEARCH BUFFER LENGTH EXCEEDS MAXIMUM %1%2%3%4**  
 The total length of the search buffer cannot exceed 4096. Use fewer IF statements or place IF selection on DEFINED fields.
- (FOC4467) **INTERNAL ERROR IN ADABAS INTERFACE PE OR MU HAS NO PARENT  
 1%2%3%4**
- (FOC4468) **USAGE AND ACTUAL FORMATS INCOMPATIBLE %1%2%3%4**  
 In the Master File, the length specified as the USAGE length for the field specified is not large enough to accommodate the values returned based on the ACTUAL field length specification.
- (FOC4469) **JOIN FIELD IS NOT A DESCRIPTOR FIELD %1%2%3%4**  
 The field indicated is specified as the cross-referenced field in the JOIN command. It is a valid Master File field, but it is not specified with the INDEX=I parameter. The field must be an ADABAS descriptor, subdescriptor, or superdescriptor.
- (FOC4470) **JOIN FIELDS HAVE INCOMPATIBLE FORMATS %1%2%3%4**
- (FOC4471) **JOIN FIELDS HAVE INCOMPATIBLE LENGTHS %1%2%3%4**
- (FOC4472) **FIELD NOT IN SEG %1%2%3%4**
- (FOC4473) **INVALID VALUE FOR FIELD TYPE %1%2%3%4**
- (FOC4474) **UNSUPPORTED TEST FOR NOP FIELD %1%2%3%4**  
 The field indicated is defined with TYPE=NOP. It was used in the table request with a screening condition that is not supported for TYPE=NOP fields. The supported relational operators for such fields are: EQ (or IS), GT, LT, GE, LE, and FROM-TO.

## Interface Error Messages

- (FOC4475) **TYPE=NOP FIELD WITH NO ALIAS USED IN IF %1%2%3%4**  
The field indicated is defined with TYPE=NOP and has no ALIAS. It was used in a screening condition in the table request. An ALIAS must be supplied so that the ADABAS Interface can use TYPE=NOP fields as part of the selection criteria for ADABAS records.
- (FOC4476) **DUPLICATE SEGMENTS IN ACCESS FILE %1%2%3%4**  
The Access File contains two SEGNAM statement lines for the same Master File segment. There is a duplicate SEGNAM for a single Master File segment or the SEGNAME value for another segment erroneously refers to this segment.
- (FOC4477) **RELEASE VALUE IS NOT SPECIFIED %1%2%3%4**  
The RELEASE parameter was specified with no value assigned to it. The release level of the ADABAS software must be specified.
- (FOC4478) **INVALID VALUE FOR DBNO %1%2%3%4**  
A value specified for the file number on a segment in the Access File is incorrect. It must be a numeric value from 1 - 256. This message is returned when the segment statement in the Access File is specified without keywords.
- (FOC4479) **EOF REACHED IN ACCESS FILE %1%2%3%4**
- (FOC4480) **INVALID SEQUENCE FIELD %1%2%3%4**  
The fieldname specified as the SEQFIELD in the Access File does not exist in the Master File or is not defined with the INDEX=I keyword.
- (FOC4481) **FIELD RECORD FOUND BEFORE SEGMENT RECORDS %1%2%3%4**
- (FOC4482) **%1%2%3%4**
- (FOC4483) **INSUFFICIENT MAIN STORAGE %1%2%3%4**  
The amount of storage available is insufficient to run the ADABAS Interface request.
- (FOC4484) **INVALID KEYWORD IN PLACE OF SEGNAME SPECIFIED %1%2%3%4**
- (FOC4485) **SEGMENT OF KEY FIELD NOT ACCESSED %1%2%3%4**
- (FOC4486) **ILLEGAL PARENT/CHILD ACCESS RELATIONSHIP %1%2%3%4**  
The relationship between the two segments indicated is not possible within an ADABAS file: either the parent segment was defined as an MU segment and the descendent as an MU or PE, or the parent and descendent were both defined as PE. These are illegal from the ADABAS point of view.
- (FOC4487) **EMPTY ALIAS FOR FIELD %1%2%3%4**  
The indicated field was defined in the Master File without an ALIAS. For FIELDS which are not part of a GROUP, the ALIAS must always be specified; it must be the two-character ADABAS fieldname.
- (FOC4488) **EMPTY ALIAS FOR GROUP AND ITS ELEMENT %1%2%3%4**  
The field indicated in the error is part of the indicated group. Neither the field nor the group was specified in the Master File with an ALIAS. For group fields, the ALIAS must be specified for the group or all the fields within the group.

- (FOC4489) **FORMAT ERROR FOR NX COUNT %1%2%3%4**  
 The ACTUAL format length is larger than the USAGE format length.
- (FOC4490) **UNSUCCESSFUL ATTEMPT TO RELEASE CID %1%2%3%4**  
 On the ADABAS Interface call to release the Command ID, an error was returned from ADABAS. The response code appears at the right. See the *Software/AG Command Reference Manual* for a description of the response code.
- (FOC4491) **FIND ERROR READING FIRST RECORD %1%2%3%4**  
 On an ADABAS Interface FIND call to retrieve records, an error was returned from ADABAS. The response code appears at the right. See the *Software/AG Command Reference Manual* for a description of the response code.
- (FOC4492) **OCCURS COUNT FIELD MUST HAVE USAGE I4 ACTUAL I1 %1%2%3%4**
- (FOC4493) **FIND ERROR READING FIRST RECORD %1%2%3%4**
- (FOC4494) %1%2%3%4
- (FOC4495) %1%2%3%4
- (FOC4496) **ERROR RETURN ON ATTEMPTING TO OPEN %1%2%3%4**  
 On an ADABAS Interface OPEN call, an error was returned from ADABAS. The response code appears at the right. See the *Software/AG Command Reference Manual* for a description of the response code.
- (FOC4497) **INVALID KEYWORD IN PLACE OF AFD SPECIFIED %1%2%3%4**
- (FOC4498) **ERROR RETURN ON READ PHYSICAL %1%2%3%4**  
 On the ADABAS Interface Read Physical call to retrieve records, an error was returned from ADABAS. The response code appears at the right. See the *Software/AG Command Reference Manual* for a description of the response code.
- (FOC4499) **ERROR IN ADABAS CLOSE %1%2%3%4**  
 On the ADABAS Interface CLOSE call, an error was returned from ADABAS. The response appears at the right. See the *Software/AG Command Reference Manual* for a description of the response code.
- (FOC4500) **READ ISN ERROR ON PE OR MU SEGMENT %1%2%3%4**  
 On the ADABAS Interface Read ISN call to retrieve records, an error was returned from ADABAS. The response code appears at the right. See the *Software/AG Command Reference Manual* for a description of the response code.
- (FOC4501) **ERROR LOADING %1%2%3%4**  
 The ADABAS Interface was unable to load an external module, most likely ADAUSER in MVS or ADABAS in VM. The program that the ADABAS Interface was attempting to load when the error occurred appears at the right.
- (FOC4502) **HISTOGRAM CALL (L9) ERROR %1%2%3%4**
- (FOC4503) **INVALID CALLTYPE FOR SEGMENT %1%2%3%4**  
 The CALLTYPE keyword in the Access File was specified on the named segment, but its value was not FIND or RL.

## Interface Error Messages

- (FOC4504) **READ LOGICAL (L3) ERROR %1%2%3%4**  
On an ADABAS Interface Read Logical call to retrieve records, an error was returned from ADABAS. The response code appears at the right. See the *Software/AG Command Reference Manual* for a description of the response code.
- (FOC4505) **ERROR RETURN ON READ ISN GET NEXT %1%2%3%4**  
On an ADABAS Interface Read ISN call to retrieve records, an error was returned from ADABAS. The response code appears at the right. See the *Software/AG Command Reference Manual* for a description of the response code.
- (FOC4506) **ERROR PASSWORD IS NOT SPECIFIED %1%2%3%4**
- (FOC4507) **ERROR DBNO IS NOT SPECIFIED %1%2%3%4**
- (FOC4508) **MAXIMUM NUMBER OF SET PASSWORDS IS REACHED**
- (FOC4509) **ERROR IN ADABAS SETTING %1%2%3%4**
- (FOC4511) **SET FOR %1%2%3%4**
- (FOC4512) **SET FETCH IS %1%2%3%4**
- (FOC4513) **SET FETCHSIZE IS %1%2%3%4**
- (FOC4514) **NO ACCESS RECORD FOR SEGMENT %1%2%3%4**  
The segment indicated in the error was specified in the Master File but has no corresponding SEGNAM statement in the Access File. The SEGNAM statement for this segment is missing or misspelled in the Access File.
- (FOC4515) **%1%2%3%4**
- (FOC4516) **NO RELEASE RECORD FOR FILE %1%2%3%4**  
The RELEASE parameter was not specified in the Access File or was misspelled. Other Access File errors can force this error message if the Access File is read beyond end-of-file.
- (FOC4517) **INVALID KEYWORD FOR DBNO SPECIFIED %1%2%3%4**
- (FOC4518) **INVALID KEYWORD FOR FILENO SPECIFIED %1%2%3%4**
- (FOC4519) **INVALID VALUE FOR FILENO %1%2%3%4**
- (FOC4520) **GROUP FIELD DEFINITION ERROR %1%2%3%4**
- (FOC4521) **%1%2%3%4**
- (FOC4522) **%1%2%3%4**
- (FOC4523) **%1%2%3%4**
- (FOC4524) **%1%2%3%4**
- (FOC4525) **INVALID KEYWORD IN ACCESS FILE %1%2%3%4**  
The keyword on the indicated line in the Access File was misspelled or is not a valid keyword for the Access File statement.

- (FOC4526) **KEYWORD VALUE EXCEEDS MAXIMUM LENGTH %1%2%3%4**  
The value indicated was specified for a keyword on the indicated line number in the Access File. The length of this value exceeds the maximum allowed for that keyword.
- (FOC4527) **DUPLICATE KEYWORDS IN ACCESS FILE RECORD %1%2%3%4**  
The line number specified in the Access File contains two values for the same keyword the second value of which is displayed with the error message.
- (FOC4528) **RECORD DOES NOT START WITH KEYWORD %1%2%3%4**  
The line number specified in the Access File does not begin with SEGNAME= or RELEASE=. These parameters must be explicitly specified along with their values.
- (FOC4529) **NO VALUE SPECIFIED FOR KEYWORD %1%2%3%4**  
An Access File with the values defined positionally has too few values.
- (FOC4530) **MISUSE OF QUOTES IN KEYWORD FILE %1%2%3%4**  
The line number indicated in the Access File contains a value for a keyword that has a single quote ('). Either the beginning or ending quote is missing or a quote was erroneously placed in the value of a keyfield.
- (FOC4531) **NO KEYWORD FOR ACCESS FILE VALUE %1%2%3%4**  
The line number indicated in the Access File contains keyword values specified positionally (without the keywords themselves). Too many such values were specified for this segment.
- (FOC4532) **INVALID ORDER FIELD FOR SEG %1%2%3%4**
- (FOC4533) **NO ALIAS SPECIFIED FOR DESCRIPTOR %1%2%3%4**
- (FOC4533) **NO ALIAS SPECIFIED FOR DESCRIPTOR %1%2%3%4**
- (FOC4534) **%1%2%3%4**
- (FOC4535) **%1%2%3%4**
- (FOC4536) **%1%2%3%4**
- (FOC4537) **%1%2%3%4**
- (FOC4538) **%1%2%3%4**
- (FOC4539) **%1%2%3%4**
- (FOC4540) **XMI TRANSMISSION FAILURE %1%2%3%4**
- (FOC4542) **XMI ACCESS FAILURE FOR SEGMENT %1%2%3%4**
- (FOC4543) **XMI TARGET SET-UP FAILURE %1%2%3%4**
- (FOC4544) **NO SPACE ON TARGET SIDE %1%2%3%4**
- (FOC4545) **RELEASE SPACE ERROR ON TARGET SIDE %1%2%3%4**
- (FOC4546) **NO AVAILABLE CIDS AT THIS TIME %1%2%3%4**
- (FOC4547) **NO AVAILABLE SESSION AT THIS TIME %1%2%3%4**

## Interface Error Messages

(FOC4548) INVALID SESSION ID %1%2%3%4  
(FOC4549) MAXIMUM NUMBER OF SET DBNOs IS REACHED %1%2%3%4  
(FOC4550) FIELD DESCRIPTIONS IN MFD AND AFD ARE MISMATCHED %1%2%3%4  
(FOC4551) INVALID VALUE FOR NU KEYWORD IN AFD %1%2%3%4  
(FOC4552) FIND (S1) ON NON DESCRIPTOR FIELD IS %1%2%3%4  
(FOC4553) INVALID VALUE FOR FETCH KEYWORD IN AFD %1%2%3%4  
(FOC4554) REQUESTED NUMBER OF RECORDS FOR FETCH BUFFER IS WRONG %1%2%3%4

## C Sample File Descriptions

---

**Topics:**

- VEHICLES Sample Files
- EMPLOYEES Sample Files
- ACUSTOMR Sample Files
- AMKTORDR Sample Files

This appendix contains the Master Files, Access Files, and (in some cases) FDTs for the databases used as examples in this manual. The files included in this appendix are views of the sample databases and do not represent the complete ADABAS database.

Master and Access Files are created by the ADABAS Interface; Software AG provides the FDTs.

### C.1 VEHICLES Sample Files

---

The following sections contain the examples of the VEHICLES Master File, Access File, and FDT.

## Sample File Descriptions

### VEHICLES Master File

---

```
$$$ CREATED BY AUTOADBS ON 12/10/97 AT 10.17.27 BY PMSMJBJ
FILENAME=ADACAR,SUFFIX=ADBSINX,$

$ ADABAS FILE = VEHICLES-FILE                                DICTIONARY = 6
SEGNAME=S01          ,SEGTYPE=S,$

FIELD= REG_NUM                ,ALIAS= AA          ,A15  ,A15  , INDEX=I,$
FIELD= CHASSIS_NUM            ,ALIAS= AB          ,I9   ,I4   , $
FIELD= PERSONNEL_ID           ,ALIAS= AC          ,A8   ,A8   , INDEX=I,$
GROUP= CAR_DETAILS            ,ALIAS= CD          ,A50  ,A50  , $
  FIELD= MAKE                  ,ALIAS= AD          ,A20  ,A20  , INDEX=I,$
  FIELD= MODEL                 ,ALIAS= AE          ,A20  ,A20  , $
  FIELD= COLOR                 ,ALIAS= AF          ,A10  ,A10  , INDEX=I,$
FIELD= YEAR                   ,ALIAS= AG          ,P2   ,Z2   , $
FIELD= CLASS                  ,ALIAS= AH          ,A1   ,A1   , INDEX=I,$
FIELD= LEASE_PUR              ,ALIAS= AI          ,A1   ,A1   , $
FIELD= DATE_ACQ              ,ALIAS= AJ          ,P6   ,Z6   , $
$GRMU = CAR_MAINTENANCE      ,ALIAS= AK          ,A11  ,A7   , $
  FIELD= CURR_CODE            ,ALIAS= AL          ,A3   ,A3   , $
  FIELD= MAINT_COST_CNT       ,ALIAS= AMC         ,I4   ,I2   , $
FIELD= DAT_ACQ_DESC           ,ALIAS= AN          ,A4   ,A4   , INDEX=I,$
GROUP= MODEL_YEAR_MAKE       ,ALIAS= AO          ,A28  ,A22  , INDEX=I,$
  FIELD=YEAR_S02              ,ALIAS= AG          ,P2   ,Z2   , $
  FIELD=MAKE_S02              ,ALIAS= AD          ,A20  ,A20  , INDEX=I,$

SEGNAME=AM0101  ,SEGTYPE=S,PARENT=S01  ,OCCURS=AMC,$  MAX= 60
FIELD= MAINT_COST                ,ALIAS= AM          ,P7   ,P4   , $
FIELD= AM0101_OCC                ,ALIAS= ORDER       ,I4   ,I2   , $
```

### VEHICLES Access File

---

```
$$$ CREATED BY AUTOADBS ON 12/10/97 AT 10.17.27 BY PMSMJBJ
$$$ FILENAME=ADACAR,SUFFIX=ADBSINX,$
RELEASE=6,OPEN=YES,$

$ ADABAS FILE = VEHICLES-FILE                                DICTIONARY = 6
SEGNAM=S01          ,ACCESS=ADBS,FILENO=002,
                      CALLTYPE=RL,SEQFIELD=PERSONNEL_ID,$
FIELD= DAT_ACQ_DESC                ,TYPE=NOP,$
FIELD= MODEL_YEAR_MAKE              ,TYPE=SPR,$
  FIELD=YEAR_S02                    ,TYPE=      ,NU=YES,$
  FIELD=MAKE_S02                    ,TYPE=DSC,NU=YES,$
SEGNAM=AM0101,ACCESS=MU  ,FILENO=002,$ MAINT_COST
```

VEHICLES FDT

```

*****
* FILE      2 (VEHICLES      ) *
*****
FIELD DESCRIPTION TABLE

```

LEVEL	I	I	I	I	I	I	I	I
	NAME	LENGTH	FORMAT	OPTIONS	PARENT OF			
1	AA	15	A	DE,UQ,NU				
1	AB	4	F	FI				
1	AC	8	A	DE				
1	CD							
2	AD	20	A	DE,NU	SUPERDE			
2	AE	20	A	NU				
2	AF	10	A	DE,NU				
1	AG	2	U	NU	SUPERDE			
1	AH	1	A	DE,FI				
1	AI	1	A	FI				
1	AJ	6	U	NU	SUPERDE			
1	AK							
2	AL	3	A	NU				
2	AM	4	P	MU,NU				

```

-----
SPECIAL DESCRIPTOR TABLE

```

TYPE	I	I	I	I	I	I	I	I
	NAME	LENGTH	FORMAT	OPTIONS	STRUCTURE			
SUPER	AN	4	B	DE,NU	AJ ( 5 - 6 )			
SUPER	AO	22	A	DE,NU	AG ( 1 - 2 )			
					AD ( 1 - 20 )			

## C.2 EMPLOYEES Sample Files

The following sections contain the examples of the EMPLOYEES Master File, Access File, and FDT.

### EMPLOYEES Master File

```

$$$ CREATED BY AUTOADBS ON 12/10/97 AT 10.09.16 BY PMSMJ
FILENAME=EMPFIL1,SUFFIX=ADBSINX,$

$ ADABAS FILE = EMPLOYEES                                DICTIONARY = 6
SEGNAME=S01      ,SEGTYPE=S,$
FIELD= PERSONNEL_ID      ,ALIAS= AA      ,A8      ,A8      , INDEX=I,$
GROUP= FULL_NAME        ,ALIAS= AB      ,A60     ,A60     ,$
  FIELD=FIRST_NAME      ,ALIAS= AC      ,A20     ,A20     ,$
  FIELD=NAME            ,ALIAS= AE      ,A20     ,A20     , INDEX=I,$
  FIELD= MIDDLE_NAME    ,ALIAS= AD      ,A20     ,A20     ,$
FIELD= MAR_STAT         ,ALIAS= AF      ,A1      ,A1      ,$
FIELD= SEX              ,ALIAS= AG      ,A1      ,A1      ,$
FIELD= BIRTH            ,ALIAS= AH      ,P6      ,Z6      , INDEX=I,$
$GRMU = FULL_ADDRESS    ,ALIAS= A1      ,A53     ,A53     ,$
  FIELD=ADDRESS_LINE_CNT,ALIAS= AIC     ,I4      ,I2      ,$
  FIELD=CITY            ,ALIAS= AJ      ,A20     ,A20     , INDEX=I,$
  FIELD=POST_CODE      ,ALIAS= AK      ,A10     ,A10     ,$
  FIELD=COUNTRY        ,ALIAS= AL      ,A3      ,A3      ,$
GROUP= TELEPHONE        ,ALIAS= A2      ,A21     ,A21     ,$
  FIELD=AREA_CODE      ,ALIAS= AN      ,A6      ,A6      ,$
  FIELD=PHONE          ,ALIAS= AM      ,A15     ,A15     ,$
FIELD= DEPT             ,ALIAS= AO      ,A6      ,A6      , INDEX=I,$
FIELD= JOB_TITLE       ,ALIAS= AP      ,A25     ,A25     , INDEX=I,$
FIELD= INCOME_CNT      ,ALIAS= AQC     ,I4      ,I2      ,$
GROUP= LEAVE_DATA      ,ALIAS= A3      ,A16     ,A4      ,$
  FIELD=LEAVE_DUE      ,ALIAS= AU      ,P2      ,Z2      ,$
  FIELD=LEAVE_TAKEN    ,ALIAS= AV      ,P2      ,Z2      ,$
FIELD= LEAVE_BOOKED_CNT,ALIAS= AWC     ,I4      ,I2      ,$
FIELD= LANG_CNT        ,ALIAS= AZC     ,I4      ,I2      ,$
$PH = PHONETIC_NAME    ,ALIAS= PH      ,A20     ,A20     ,
$
  INDEX=I,$

```

## The EMPLOYEES Master File continued:

```

GROUP= LEAVE_LEFT           ,ALIAS= H1           ,A16 ,A4 , INDEX=I,$
  FIELD=LEAVE_DUE_S01       ,ALIAS= AU           ,P2  ,Z2 , $
  FIELD=LEAVE_TAKEN_S01     ,ALIAS= AV           ,P2  ,Z2 , $
FIELD= DEPARTMENT           ,ALIAS= S1           ,A4  ,A4 , INDEX=I,$
GROUP= DEPT_PERSON          ,ALIAS= S2           ,A26 ,A26 , INDEX=I,$
  FIELD=DEPT_S03            ,ALIAS= AO           ,A6  ,A6 , INDEX=I,$
  FIELD=NAME_S03           ,ALIAS= AE           ,A20 ,A20 , INDEX=I,$

SEGNAME=AI0101 ,SEGTYPE=S,PARENT=S01 ,OCCURS=AIC,$ MAX= 8
  FIELD= ADDRESS_LINE       ,ALIAS= AI           ,A20 ,A20 , $
  FIELD= AI0101_OCC         ,ALIAS= ORDER        ,I4  ,I2 , $

SEGNAME=AQ0201 ,SEGTYPE=S,PARENT=S01 ,OCCURS=AQC,$ MAX= 40
$PEMU = INCOME             ,ALIAS= AQ           ,A19 ,A13 , $
  FIELD=CURR_CODE           ,ALIAS= AR           ,A3  ,A3 , $
  FIELD=SALARY              ,ALIAS= AS           ,P9  ,P5 , $
  FIELD=BONUS_CNT           ,ALIAS= ATC          ,I4  ,I2 , $
  FIELD= AQ0201_OCC         ,ALIAS= ORDER        ,I4  ,I2 , $

SEGNAME=AT0301 ,SEGTYPE=S,PARENT=AQ0201,OCCURS=ATC,$ MAX= 12
  FIELD= BONUS              ,ALIAS= AT           ,P9  ,P5 , $
  FIELD= AT0301_OCC         ,ALIAS= ORDER        ,I4  ,I2 , $

SEGNAME=AW0401 ,SEGTYPE=S,PARENT=S01 ,OCCURS=AWC,$ MAX= 20
GROUP= LEAVE_BOOKED        ,ALIAS= AW           ,A16 ,A12 , $
  FIELD=LEAVE_START         ,ALIAS= AX           ,P6  ,Z6 , $
  FIELD=LEAVE_END           ,ALIAS= AY           ,P6  ,Z6 , $
  FIELD= AW0401_OCC         ,ALIAS= ORDER        ,I4  ,I2 , $

SEGNAME=AZ0501 ,SEGTYPE=S,PARENT=S01 ,OCCURS=AZC,$ MAX= 15
  FIELD= LANG               ,ALIAS= AZ           ,A3  ,A3 , INDEX=I,$
  FIELD= AZ0501_OCC         ,ALIAS= ORDER        ,I4  ,I2 , $

```

## EMPLOYEES Access File

---

```
$$$ CREATED BY AUTOADBS ON 12/10/97 AT 10.09.16 BY PMSMJB
$$$ FILENAME=EMPFIL1,SUFFIX=ADBSINK,$
RELEASE=6,OPEN=YES,$

$ ADABAS FILE = EMPLOYEES                                DICTIONARY = 6
SEGNAM=S01 ,ACCESS=ADBS,FILENO=001,
      CALLTYPE=RL,SEQFIELD=PERSONNEL_ID,$
FIELD= LEAVE_LEFT ,TYPE=SPR,$
FIELD=LEAVE_DUE_S01 ,TYPE= ,NU=NO,$
FIELD=LEAVE_TAKEN_S01 ,TYPE= ,NU=YES,$
FIELD= DEPARTMENT ,TYPE=NOP,$
FIELD= DEPT_PERSON ,TYPE=SPR,$
FIELD=DEPT_S03 ,TYPE=DSC,NU=NO,$
FIELD=NAME_S03 ,TYPE=DSC,NU=NO,$
SEGNAM=AI0101,ACCESS=MU ,FILENO=001,$ ADDRESS_LINE
SEGNAM=AQ0201,ACCESS=PE ,FILENO=001,$ INCOME
SEGNAM=AT0301,ACCESS=MU ,FILENO=001,$ BONUS
SEGNAM=AW0401,ACCESS=PE ,FILENO=001,$ LEAVE_BOOKED
SEGNAM=AZ0501,ACCESS=MU ,FILENO=001,$ LANG
```

The EMPLOYEES FDT

```

*****
* FILE      1 (EMPLOYEES      ) *
*****
FIELD DESCRIPTION TABLE

```

LEVEL	I	I	I	I	I	I	I	I
	NAME	LENGTH	FORMAT	OPTIONS	PARENT OF			
1	AA	8	A	DE,UQ				
1	AB							
2	AC	20	A	NU				
2	AE	20	A	DE	PHONDE ,SUPERDE			
2	AD	20	A	NU				
1	AF	1	A	FI				
1	AG	1	A	FI				
1	AH	6	U	DE				
1	A1							
2	AI	20	A	MU,NU				
2	AJ	20	A	DE,NU				
2	AK	10	A	NU				
2	AL	3	A	NU				
1	A2							
2	AN	6	A	NU				
2	AM	15	A	NU				
1	AO	6	A	DE	SUBDE ,SUPERDE			
1	AP	25	A	DE,NU				
1	AQ			PE				
2	AR	3	A	NU	SUPERDE			
2	AS	5	P	NU	SUPERDE			
2	AT	5	P	MU,NU				
1	A3							
2	AU	2	U		SUPERDE			
2	AV	2	U	NU	SUPERDE			
1	AW			PE				
2	AX	6	A	NU				
2	AY	6	P	NU				
1	AZ	3	P	DE,MU,NU				

```

-----
SPECIAL DESCRIPTOR TABLE

```

TYPE	I	I	I	I	I	I	I	I
	NAME	LENGTH	FORMAT	OPTIONS	STRUCTURE			
SUPER	H1	4	B	DE,NU	I AU ( 1 - 2)			
					I AV ( 1 - 2)			
SUB	S1	4	A	DE	I AO ( 1 - 4)			
SUPER	S2	26	A	DE	I AO ( 1 - 6)			
					I AE ( 1 - 20)			
SUPER	S3	12	A	DE,NU,PE	I AR ( 1 - 3)			
					I AS ( 1 - 9)			
					I			
PHON	PH				I PH = PHON (AE)			

## C.3 ACUSTOMR Sample Files

The following sections contain the examples of the ACUSTOMR Master and Access Files.

### ACUSTOMR Master File

```

FILE=ACUSTOMR          , SUFFIX=ADBSINX
SEGNAME=CUSTOMER
FIELDNAME  =NCUSTOMR_GRP      , AA          , I6          , I2,      INDEX=I , $
FIELDNAME  =NAMECUST         , AB          , A15         , A15     , $
FIELDNAME  =DCUSROAD         , AC          , A20         , A20     , $
FIELDNAME  =DCUSTOWN         , AD          , A20         , A20     , $
    
```

### ACUSTOMR Access File

```

RELEASE=6.0 , OPEN=YES , $
SEGNAM=CUSTOMER , ACCESS=ADBS , FILENO=021 , DBNO=001 , CALLTYPE=RL , $
    
```

## C.4 AMKTORDR Sample Files

The following sections contain the examples of the AMKTORDR Master and Access Files.

### AMKTORDR Master File

```

FILE=AMKTORDR          , SUFFIX=ADBSINX
SEGNAME=MKTORDER
FIELDNAME  =NMARKET_GRP      , BA          , I3          , I2,      INDEX=I , $
FIELDNAME  =QPRODUCT         , BB          , I3          , I2     , $
FIELDNAME  =QNEEDITM        , BC          , I3          , I2     , $
FIELDNAME  =FBUILD          , BD          , A1          , A1     , $
FIELDNAME  =FK_NPRODUCT     , BE          , A4          , A4     , $
FIELDNAME  =FK_NCUSTOMER    , BF          , I3          , I2     , $
FIELDNAME  =DATEMKTO        , BG          , A8          , A8     , $
FIELDNAME  =DATEFBLD        , BH          , A8          , A8     , $
    
```

### AMKTORDR Access File

```

RELEASE=6.0 , OPEN=YES , $
SEGNAM=MKTORDER , ACCESS=ADBS , FILENO=022 , DBNO=001 , $
    
```

# Index

---

---

## Numbers

---

24-bit Mode, 8-7

## A

---

### ACCESS

- Ddname, A-8
- Filetype, A-19

ACCESS attribute, 4-23, 5-22, 6-7  
ADBS, 5-23

Access File, 2-6, 3-7. *See also* Field declaration and Segment declaration

- As PDS member, 5-1, 5-18, A-8
- Attributes, 4-23, 5-18
- AUTOADBS, 4-9, 4-22
  - Attributes Screen, 4-12
- Changing generated descriptions, 4-28
- Comments, 4-2
- Default name, A-14
- Describing to FOCUS, 5-1
- FETCH syntax, 8-3
- FETCHSIZE syntax, 8-3
- Filetype, 5-1, 5-18, A-19
- Read Logical navigation, 7-4
- Release declaration syntax, 5-19

### ACTUAL

- Attribute, 4-23, 5-6, 5-9, 5-10
- AUTOADBS format, 4-27

ACUSTOMR, C-8

### ADABAS

- Environment, 3-1, 3-2
- Fieldname, 3-4
- Fields, reformatting with ALIAS, 5-7
- Response codes, 9-2, A-9

ADABAS text library, 2-5

ADABAS.DATA, A-4, A-9

ADABAS.LOAD, A-4, A-8

### ADAUSER

- CMS, A-18
- MVS, A-7

ADBS\$PRM, 4-7

ADBSINX, 4-22, 5-1, 5-3, A-4

Address Converter, 3-3

ALIAS attribute, 4-23, 4-26, 5-6, 5-7

Aliases in AUTOADBS, 4-26

ALL prefix, 6-4

AMKTORDR, C-8

Associator components, 3-3

AUTOADBL ddname (MVS), 4-27

AUTOADBL FOCTEMP (CMS), 4-29

AUTOADBS, 3-7, 4-1

- Access File
  - Attributes, 4-23
  - Generating, 4-9
- Access File Attributes Screen, 4-12
- ACTUAL format, 4-27
- Aliases, 4-26
- Background execution, 4-20
- Child selection, 4-14, 4-15
- Child Selection Screen, 4-14
- Executing, 4-3
- Fieldnames, 4-25
- File Selection Screen, 4-9
- Foreign keys, 4-14, 4-16
- Generated descriptions, 4-22
  - Changing, 4-28
  - List of, 4-28
- Installation
  - CMS, A-19
  - MVS, A-4
  - Verifying, 9-9

## Index

### AUTOADBS (*continued*)

- IXFLD Selection Screen, 4-16
- KEYFLD Selection Screen, 4-18
- Main Menu, 4-5
- Master File
  - Attributes, 4-22, 4-23
  - Generating, 4-9
- Parameter log file, 4-7, 4-30
- PF keys, 4-7, 4-10, 4-13, 4-15, 4-17, 4-19
- Primary keys, 4-14, 4-18
- Required files, 4-3
- USAGE format, 4-27
- User views, 4-10

## B

---

Background execution of AUTOADBS, 4-20

Batch access, 2-4

Batch Region, 3-2

## C

---

CALLTYPE attribute, 4-11, 4-23, 5-21, 5-24, A-13, A-22

CALLTYPE=FINN retrieval, 6-6, 8-7

CALLTYPE=RL

- Optimization, 6-8
- Retrieval, 7-6

Child files in AUTOADBS, 4-14

CLIST, 2-2, A-15

### CMS

- Access File, 5-1, 5-18
- AUTOADBL FOCTEMP, 4-29
- AUTOADBS execution, 4-3
- Interface installation, A-1, A-17
- Interface interactive access, 2-5
- Master File, 5-1

CNT field in AUTOADBS, 4-25. *See also* Counter fields

Commands, environment, 8-1

Comments in Access and Master Files, 4-2

Complex FIND calls

- Data retrieval, 7-7
- Descendant records, 7-12

Concatenated datasets, 2-4

Counter fields

- AUTOADBS, 4-25
- Repeating fields, 3-12, 3-14, 3-15, 5-15, 5-16, 5-17

Cross-reference

- Attributes, 5-29
- Fields in AUTOADBS, 4-25
- With GROUP, 5-26

## D

---

Data Definition Module (DDM), 3-3

- External field name, 3-5

Data dictionary, 3-3. *See also* Predict dictionary

Data integrity, 6-8, 7-5

Data retrieval, 3-1, 3-3

- Calls
  - Complex FIND, 7-7, 7-12
  - L9 direct, 7-8
  - Read Logical, 7-4, 7-5, 7-10
  - Read Physical, 7-3
  - Simple FIND, 7-6, 7-11
- Descendant ADABAS records, 7-9
- Descendant periodic groups and multi-value fields, 7-9
- Entry segment, 7-1
- GET FIRST, 7-7
- GET NEXT, 7-7
- Methods, 7-1
- Null suppression, 5-29, 7-5
- READLIMIT, 6-5
- RECORDLIMIT, 6-5
- SEQFIELD, 7-4, 7-5, 7-8

Data Storage Area, 3-1, 3-2

Data storage management, 3-6

Dataset default names, A-14

DBNO attribute, 4-22, 5-23, 8-3, A-3

DBNR, 4-12

DCB parameters, A-8

DDCARD, 2-3, 2-4, A-3

DDM. *See* Data Definition Module

Debugging techniques, 9-1
 

- Trace facility, 9-2
  - FSTRACE, 9-3
  - FSTRACE4, 9-3
  - FSTRACE5, 9-3
- Invoking, 9-7
- Turning off, 9-7
- Troubleshooting, 9-9

Default parameters in AUTOADBS, 4-7

DEFINEd fields, 4-25

Descendant periodic groups and multi-value fields, 7-9

Descendant records, 7-9
 

- Complex FIND calls, 7-12
- Read Logical calls, 7-10
- Simple FIND calls, 7-11

Descriptors, 3-4
 

- Hyper, 4-24
- JOIN to, 6-7
- Phonetic, 4-24
- Status, 3-5
- Type, 3-9

Disk space requirements for AUTOADBS, 4-3

Distribution tape
 

- Contents, A-4, A-18
- Identifying release level, A-2

Dynamic Area, 3-2

Dynamic database number (SET DBNO), 8-3

## E

---

EMPLOYEEES
 

- Access File, C-6
- FDT, C-7
- Master File, C-4

Environment commands, 8-1

Error messages, 9-2, B-1
 

- Library, A-9
- Response codes, 9-2

ERRORS ddname, A-9

ERRORS.DATA, A-9

EX AUTOADBS BATCH, 4-20

EX AUTOADBS MFDLIST, 4-29

Examples
 

- Access File
  - ACUSTOMR, C-8
  - AMKTORDR, C-8
  - EMPLOYEEES, C-6
  - VEHICLES, C-2
- FDT
  - EMPLOYEEES, C-7
  - VEHICLES, C-3
- Master File
  - ACUSTOMR, C-8
  - AMKTORDR, C-8
  - EMPLOYEEES, C-4
  - VEHICLES, C-1

External field name, 3-5

## F

---

Features in AUTOADBS, 4-1

FETCH attribute, 5-22, 5-25
 

- Access File syntax, 8-3
- SET command, 8-2

FETCHSIZE attribute, 5-22, 5-25
 

- Access File syntax, 8-3
- SET command, 8-2

## Index

### Field

- Properties, 3-5
- Suffix, 4-23, 4-24, 5-27
- Type, 3-5

### Field declaration, 5-6

- Access File
  - FIELDNAME, 4-23, 5-27
  - NU, 4-23
  - TYPE, 4-23, 5-27
- Master File
  - ACTUAL, 4-23, 5-9
  - ALIAS, 4-23, 5-7
  - FIELDNAME, 4-23
  - GROUP, 4-24, 5-12, 5-26
  - INDEX, 4-24, 5-10
  - TITLE, 4-24
  - USAGE, 4-23, 5-8

### Field Definition Table, 3-4, 3-5, 3-6, 3-9

### Field level indicator, 3-4

### FIELDNAME attribute, 4-23, 4-25, 5-6, 5-27

### Fieldname in AUTOADBS, 4-23, 4-25

### File

- ADABAS, 3-7, 3-9, 3-10
- Descriptions (FOCUS), 1-1, 2-6, 4-22, 5-1, 5-18
- Fixed-length records, 3-11
- Generated lists in AUTOADBS, 4-29
- Navigation, 6-1
- Variable-length records, 3-11

### File declaration

- FILENAME, 4-21, 5-3
- SUFFIX, 4-21, 5-3

### File descriptions, 2-6

### File Selection Screen, 4-9

### FILENAME attribute, 4-22, 5-3

### FILENO attribute, 4-23, 5-23

### FIND calls, 7-7, 7-11, 7-12

### Fixed-length record, 3-11

### FOCADBS

- Ddname, A-8
- Filetype, A-19

### FOCDEF file default name, A-14

### FOCEXEC, A-8

### FOCUS

- Installation requirement, A-2
- Procedures, 2-6
- PTF, A-2
- Release level, A-2
- Supported features, 1-1
- TableTalk, 4-1

### Foreground Region, 3-2

### Foreign key, 4-14, 4-16

### FSTRACE, 9-2

- Example, 9-4

### FSTRACE4, 9-4

- Example, 9-5

### FSTRACE5, 9-3

- Example, 9-6

## G

---

### GENFADLA member, A-11

### GET FIRST in data retrieval, 7-7

### GET NEXT in data retrieval, 7-7

### GLOBAL command, A-18

### GROUP attribute, 4-24, 5-12

- Cross-referencing files, 5-26

### Group field

- Optimizing, 6-9
- Testing with numerics, 6-10

## H

---

### Hierarchical structure, 3-8

### Hyper descriptors, 4-24

**I**


---

IEBCOPY procedure in installation, A-5

INDEX attribute, 4-23, 5-10

Installation, A-1, A-17. *See also* Libraries

- AUTOADBS
  - CMS, A-19
  - MVS, A-10
- Distribution tape, A-4, A-18
- IEBCOPY procedure, A-5
- Maintenance, A-2
- Requirements
  - Memory, A-2
  - Software, A-2
- Run-time libraries, A-7
- Site defaults using DDCARD, A-3
- Testing
  - CMS, A-23
  - MVS, A-15
- Verifying, 9-9

Interactive access

- CMS, 2-5
- MSO, 2-3
- TSO, 2-2

Internal Sequence Numbers, 3-3

Inverted lists

- Associator component, 3-3
- Types, 3-4

ISPF, A-7

IXFLD attribute, 4-23, 5-29

IXFLD Selection Screen, 4-16

**J**


---

JOIN, 6-7

- Embedded, 5-29
  - Rules, 5-31
- From, 6-7
- Multi-field, 6-8
- Short-to-long, 6-8
- To, 6-7

JOIN from, 6-7

JOIN to, 6-7

**K**


---

KEYFLD attribute, 4-23, 5-29

KEYFLD Selection Screen, 4-18

**L**


---

L9 direct calls, 7-8

Libraries (CMS)

- ADABAS text, 2-5
- Interface load, A-17
- Macro, A-18

Libraries (MSO), 2-3, A-10

Libraries (MVS)

- ACCESS.DATA, A-7
- ADABAS.DATA, A-4, A-9, A-10
- ADABAS.LOAD, 2-3, A-4, A-8, A-10
- ERRORS.DATA, A-9, A-10
- FOCADBS.DATA, 4-7, A-8
- FOCEXEC.DATA, A-8
- FOCLIB.LOAD, 2-2, A-7, A-10
- Interface load, A-6
- MASTER.DATA, A-7
- SOFTWARE.AG.ADARUN, 2-1, A-2
- SOFTWARE.AG.LOAD, A-7, A-10

## M

---

- Macro library (CMS), A-18
- Main Menu in AUTOADBS, 4-5
- Maintenance, A-2
- MASTER ddname, 5-1, A-7
- Master File, 2-6, 3-7. *See also* File declaration, Field declaration, and Segment declaration
  - As PDS member, 5-2, A-7
  - Attributes, 4-9, 4-22, 4-23, 5-3
  - Changing generated descriptions, 4-28
  - Comments, 4-2
  - Default name, A-14
  - Describing superdescriptors, 5-11
  - Filetype, 5-2
  - Periodic group with multi-value field, 5-16
- MASTER library, A-7
- MAX attribute, 4-22
- Memory requirements (installation), A-2
- MFDLIST, 4-29
- Missing segment, 6-2
- MSO
  - Installation, A-1, A-10
  - Interactive access, 2-3
- Multifetch, 8-1
- Multi-field JOIN, 6-8
- Multiple occurrences of segments, 3-8
- Multi-value fields, 3-11, 5-16
  - Data retrieval, 7-9
  - ORDER field, 5-17
  - Within periodic groups, 5-16

## MVS

- Access File, 5-1, 5-18
- AUTOADBS execution, 4-3
- Interface
  - Batch access under TSO, 2-4
  - Interactive access from MSO, 2-3
  - Interactive access from TSO, 2-2
- Interface installation, A-1, A-4
  - Testing, A-15
- ISPF, A-7
- Master File, 5-1
- MFDLIST, 4-29
- Parameter log file search order, 4-30
- Verifying environment, 9-9

## N

---

- NATURAL column heading, 4-1, 4-6
- Navigational logic, 6-1
- NU attribute, 4-23, 5-29
- NUCXTNNTS EXEC, A-19
- Null-suppression, 3-6, 4-23
  - Data retrieval, 7-5
  - Defining, 5-29

## O

---

- OCCURS attribute, 3-12, 3-14, 5-14
  - AUTOADBS, 4-22, 4-25
  - ORDER field, 5-17
- Online help in AUTOADBS, 4-1
- Online Region, 3-2
- OPEN attribute, 4-23, 5-20
- Operating environment, 1-2
- Operating system component, 3-2
- Optimization
  - FIND call, 6-6, 8-7
  - On group fields, 6-9
  - With null-suppression, 6-8

ORDER field, 3-15, 5-17

AUTOADBS, 4-25

Overview of ADABAS, 3-1

## **P**

---

PARENT attribute, 4-22, 5-4, 5-5

Parent-child segments, 3-8

Partial field, 5-28

PASS attribute, 4-23, 5-21, 5-25

Password, 4-12, 8-5

Periodic groups, 3-11, 5-16

Multi-value field, 5-16

ORDER field, 5-17

Phonetic descriptors, 4-24

PREDDDB

CMS, A-21

MVS, A-11

PREDEL

CMS, A-21

MVS, A-11

Predict dictionary, 3-3, 4-1

CMS, A-20

MVS, A-11

Prefetch, 8-1

Preparing run-time libraries, A-7

Primary key, 4-14, 4-18

Program temporary fix (PTF), A-2

## **R**

---

Read Logical calls, 7-4, 7-5, 7-11

Read Physical calls, 7-3

READLIMIT, 6-5

Record

Length, 3-10, 3-11

Retrieval for subtree, 6-1

RECORDLIMIT, 6-5

Reformat ADABAS fields, 5-7

RELEASE attribute, 4-23, 5-20

Release declaration, 4-23, 5-19

Release level, A-2

Repeating fields, 3-11, 5-14

Repeating groups, 3-11, 5-14

Report considerations, 6-1

Response codes, 9-2, A-8

Retrieval methodology for segments, 6-2

Retrieval sequence for segments, 3-8

Run-time libraries, A-7

## **S**

---

Search order for parameter logs, 4-30

Security, 1-2

Segment, 3-8

Missing, 6-2

Retrieval methodology, 6-2

Segment declaration, 5-4, 5-21

Access File

ACCESS, 4-23, 5-22

CALLTYPE, 4-23, 5-24

DBNO, 4-23, 5-23

FETCH, 5-25

FETCHSIZE, 5-25

FILENO, 4-23, 5-23

IXFLD, 4-23, 5-30

KEYFLD, 4-23, 5-30

PASS, 4-23, 5-25

SEGNAM, 4-23, 5-22

SEQFIELD, 4-23, 5-25

## Index

### Segment declaration (*continued*)

#### Master File

OCCURS, 4-22, 5-14

PARENT, 4-22, 5-4, 5-5

SEGNAME, 4-22, 5-4

SEGTYPE, 4-22, 5-4, 5-5

SEGNAM attribute, 4-22, 5-21, 5-22

SEGNAME attribute, 4-22, 5-4

SEGTYPE attribute, 4-22, 5-4, 5-5, 7-8

### Selecting

Foreign keys in AUTOADBS, 4-16

Primary keys in AUTOADBS, 4-18

Selection considerations, 6-5

### SEQFIELD

Attribute, 4-22, 5-25

AUTOADBS, 4-11

Data retrieval, 7-4, 7-8

Default, A-14, A-23

### SET

ALL, 6-3

AMODE 24 command, 8-7

DBNO, 8-3

FETCH, 8-1

FETCHSIZE, 8-1

PASSWORD, 8-5

SET ALL, 6-3

SET AMODE 24 command, 8-7

SET DBNO, 8-3

SET FETCH, 8-2

SET FETCHSIZE, 8-2

SET PASSWORD, 8-5

Short-to-long JOIN, 6-8

Simple FIND calls, 7-6, 7-11

SOFTWARE.AG.ADARUN, 2-1, A-2

SOFTWARE.AG.LOAD, A-7

Space Allocation Tables, 3-3

Standard format, 3-4, 3-5

Standard length, 3-4

Subdescriptors, 3-4, 3-9

Field suffix, 5-28, 5-31

JOIN to, 6-7

Subtree, 6-1

SUFFIX attribute, 4-22, 5-3

Suffix syntax, 5-27

SUM CNT in L9 direct calls, 7-8

Superdescriptor, 3-9

Describing, 5-11

Field suffix, 5-27

JOIN to, 6-7

Partial field, 5-28

SVC number, 2-3, A-3, A-10

## T

---

TITLE attribute, 4-24

Trace facility, 9-2

Invoking, 9-7

Turning off, 9-7

Troubleshooting, 9-9

### TSO

Installation, A-1, A-4

Interactive access, 2-2

TYPE attribute, 4-23, 5-27

## U

---

USAGE attribute, 4-23, 5-6, 5-8

USAGE in AUTOADBS, 4-27

User views in AUTOADBS, 4-10

**V**

---

Variable-length record, 3-11

**VEHICLES**

Access File, C-2

FDT, C-3

Master File, C-1

VOL=SER value, A-5

**W**

---

Work area, 3-3

Write access, 1-1

# Reader Comments

---

---

In an ongoing effort to produce effective documentation, the Documentation Services staff at Information Builders welcomes any opinion you can offer regarding this manual.

Please use this form to relay suggestions for improving this publication or to alert us to corrections. Identify specific pages where applicable. Send comments to:

Corporate Publications  
Attn: Manager of Documentation Services  
Information Builders  
Two Penn Plaza  
New York, NY 10121-2898

or FAX this page to (212) 967-0460, or call **Josephine Moscato** at (212) 736-4433, **x3670**.

Name: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

Telephone: \_\_\_\_\_ Date: \_\_\_\_\_

Comments:

# Reader Comments

---

---