



Operations

Adabas Version 7.4.2



This document applies to Adabas Version 7.4.2 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

© Copyright Software AG 1999-2003.
All rights reserved.

The name Software AG and/or all Software AG product names are either trademarks or registered trademarks of Software AG. Other company and product names mentioned herein may be trademarks of their respective owners.

Table of Contents

About this Documentation	1
About this Documentation	1
Conventions	3
Conventions	3
Syntax Conventions	3
Dataset Names	5
Controlling the Adabas Environment	6
Controlling the Adabas Environment	6
Adabas Initialization (ADARUN Statement)	8
Adabas Initialization (ADARUN Statement)	8
ADARUN Parameter Directory	9
ADARUN Parameter Directory	9
ADARUN Parameter Syntax	14
ADARUN Parameter Syntax	14
Setting Adabas Session Parameters	15
Setting Adabas Session Parameters	15
General Guidelines	15
Using Session Statistics to Determine Parameter Settings	20
Parameters for Operating Systems	20
AOSLOG : Log Update Commands from AOS and ADADBS	21
AOSLOG : Log Update Commands from AOS and ADADBS	21
AREXCLUDE : Exclude Files from Autorestart Processing	22
AREXCLUDE : Exclude Files from Autorestart Processing	22
ARMNAME : Activate ARM	23
ARMNAME : Activate ARM	23
ASSOCACHE : Associator Controller Caching	24
ASSOCACHE : Associator Controller Caching	24
ASYTVS : Asynchronous Bufferflush	25
ASYTVS : Asynchronous Bufferflush	25
CACHE : Adabas Caching Facility Control	26
CACHE : Adabas Caching Facility Control	26
CDXnn : Collation Descriptor User Exit	27
CDXnn : Collation Descriptor User Exit	27
CLOGDEV : Multiple Command Log Device Type	28
CLOGDEV : Multiple Command Log Device Type	28
CLOGSIZE : Multiple Command Log Size	28
CLOGSIZE : Multiple Command Log Size	28
CLOGLAYOUT : Command Logging Format	29
CLOGLAYOUT : Command Logging Format	29
CLOGMRG : Cluster Command Log Merge	30
CLOGMRG : Cluster Command Log Merge	30
Parameter Dependencies	30
Dynamic Modification	30
CLUCACHENAME : Cluster Cache Structure/Area Name	32
CLUCACHENAME : Cluster Cache Structure/Area Name	32
CLUCACHESIZE : Size of the Global Cache Area	33
CLUCACHESIZE : Size of the Global Cache Area	33
CLUGROUPNAME : Cluster Group Name	35
CLUGROUPNAME : Cluster Group Name	35

CLULOCKNAME : Cluster Lock Structure/Area Name	36
CLULOCKNAME : Cluster Lock Structure/Area Name	36
CLULOCKSIZE : Size of the Global Lock Area	37
CLULOCKSIZE : Size of the Global Lock Area	37
CLUSTER : Adabas Nucleus Cluster Session Control	39
CLUSTER : Adabas Nucleus Cluster Session Control	39
CMADDR and CMLADDR : GETMAIN Memory Pool Start Address	40
CMADDR and CMLADDR : GETMAIN Memory Pool Start Address	40
CMDQMODE : Command Queue Mode	41
CMDQMODE : Command Queue Mode	41
CMFIX or CMLFIX : GETMAIN Memory Pool Fixed Location	42
CMFIX or CMLFIX : GETMAIN Memory Pool Fixed Location	42
CMLxxxxx Parameters	43
CMLxxxxx Parameters	43
CMSCOPE or CMLSCOPE : GETMAIN Memory Pool Scope	44
CMSCOPE or CMLSCOPE : GETMAIN Memory Pool Scope	44
CMSIZE or CMLSIZE : GETMAIN Memory Pool Use and Size	45
CMSIZE or CMLSIZE : GETMAIN Memory Pool Use and Size	45
CT : Command Timeout Limit	46
CT : Command Timeout Limit	46
DATACACHE : Data Storage Controller Caching	47
DATACACHE : Data Storage Controller Caching	47
DBID : Database Identification	48
DBID : Database Identification	48
DEVICE : Device Type	49
DEVICE : Device Type	49
DIRRATIO / ELEMENTRATIO : Cluster Cache Directory to Elements Ratio	50
DIRRATIO / ELEMENTRATIO : Cluster Cache Directory to Elements Ratio	50
DSF : Adabas Delta Save Facility Control	51
DSF : Adabas Delta Save Facility Control	51
DSFEX1 : Delta Save Facility Exit	52
DSFEX1 : Delta Save Facility Exit	52
DTP : Distributed Transaction Processing Control	53
DTP : Distributed Transaction Processing Control	53
DUALCLD : Dual Command Log Device Type DUALCLS : Dual Command Log Size	55
DUALCLD : Dual Command Log Device Type DUALCLS : Dual Command Log Size	55
DUALPLD : Dual Protection Log Device Type DUALPLS : Dual Protection Log Size	56
DUALPLD : Dual Protection Log Device Type DUALPLS : Dual Protection Log Size	56
FASTPATH : Adabas Fastpath Control	57
FASTPATH : Adabas Fastpath Control	57
FMXIO : Parallel LFIOP I/O Operations	58
FMXIO : Parallel LFIOP I/O Operations	58
Parameter Dependencies	58
Dynamic Modification	58
FORCE : Allow Database ID Table Entry Overwrite	59
FORCE : Allow Database ID Table Entry Overwrite	59
GROUPS : User Group Interprocess Communication	60
GROUPS : User Group Interprocess Communication	60
HEXnn :Hyperdescriptor User Exit	61
HEXnn :Hyperdescriptor User Exit	61

IDTNAME : Define ID Table Name	62
IDTNAME : Define ID Table Name	62
IGNDIB : Ignore DIB Entry for Adabas Nucleus	63
IGNDIB : Ignore DIB Entry for Adabas Nucleus	63
IGNDTP : Ignore Two-Phase Commit Area (Work Part 4)	64
IGNDTP : Ignore Two-Phase Commit Area (Work Part 4)	64
INTNAS : Time Interval Between SYNS 60 Checkpoints	65
INTNAS : Time Interval Between SYNS 60 Checkpoints	65
LBP : Length of Buffer Pool	66
LBP : Length of Buffer Pool	66
LCP : Length of Security Pool	67
LCP : Length of Security Pool	67
LDEUQP : Length of Unique Descriptor Pool	68
LDEUQP : Length of Unique Descriptor Pool	68
LDTP : Length of Distributed Transaction Processing Area (Work Part 4)	69
LDTP : Length of Distributed Transaction Processing Area (Work Part 4)	69
LFIOP : Asynchronous Buffer Pool Control	70
LFIOP : Asynchronous Buffer Pool Control	70
LFP : Length of Internal Format Buffer Pool	71
LFP : Length of Internal Format Buffer Pool	71
LI : Length of Table of ISNs (TBI)	72
LI : Length of Table of ISNs (TBI)	72
LOCAL : Local Nucleus	73
LOCAL : Local Nucleus	73
LOGGING / LOGxx : Command Logging Control	74
LOGGING / LOGxx : Command Logging Control	74
LOGSIZE : Maximum Command Log Size	75
LOGSIZE : Maximum Command Log Size	75
LP : Length of Data Protection Area (Work Part 1)	76
LP : Length of Data Protection Area (Work Part 1)	76
LQ : Length of Sequential Command Table	77
LQ : Length of Sequential Command Table	77
LRDP : Length of the Cluster Block Update Redo Pool	78
LRDP : Length of the Cluster Block Update Redo Pool	78
Parameter Dependencies	78
Pool Usage Reports	78
LS : Length of Internal Sort Area	79
LS : Length of Internal Sort Area	79
LU : Length of Intermediate User Buffer Area	81
LU : Length of Intermediate User Buffer Area	81
LWKP2 : Length of ISN List Processing Area (Work Part 2)	82
LWKP2 : Length of ISN List Processing Area (Work Part 2)	82
LWP : Length of Adabas Work Pool	83
LWP : Length of Adabas Work Pool	83
MODE : Mode of Operation	84
MODE : Mode of Operation	84
MSGBUF : Size of Message Buffer	85
MSGBUF : Size of Message Buffer	85
MSGCONSL : Case of Messages Sent to the Console	
MSGDRUCK : Case of Messages Sent to DD/DRUCK	
MSGPRINT : Case of Messages Sent to DD/PRINT	86
MSGCONSL : Case of Messages Sent to the Console	
MSGDRUCK : Case of Messages Sent to	

DD/DRUCK MSGPRINT : Case of Messages Sent to DD/PRINT	86
MXMSG-Timeout Threshold for Internucleus Communication	87
MXMSG-Timeout Threshold for Internucleus Communication	87
MXTNA : Maximum Non-activity Time	88
MXTNA : Maximum Non-activity Time	88
MXTSX : Maximum Sx Command Execution Time	89
MXTSX : Maximum Sx Command Execution Time	89
MXTT : Maximum Transaction Time	90
MXTT : Maximum Transaction Time	90
NAB : Number of Attached Buffers	91
NAB : Number of Attached Buffers	91
NC : Number of Command Queue Elements	92
NC : Number of Command Queue Elements	92
NCLOG : Number of Command Log Datasets	93
NCLOG : Number of Command Log Datasets	93
NH : Number of Hold Queue Elements	94
NH : Number of Hold Queue Elements	94
NISNHQ : Number of ISNs in Hold Queue per User	95
NISNHQ : Number of ISNs in Hold Queue per User	95
Overriding the Parameter Setting	95
NONDES : Nondesoriptor Searches	96
NONDES : Nondesoriptor Searches	96
NPLOG : Number of Protection Log Datasets	97
NPLOG : Number of Protection Log Datasets	97
NQCID : Number of Active Command IDs per User	98
NQCID : Number of Active Command IDs per User	98
Overriding the Parameter Setting	98
NSISN : Number of ISNs per ISN Table Element	99
NSISN : Number of ISNs per ISN Table Element	99
IBM and Compatible Devices	99
BS2000 Devices	100
Overriding the Parameter Setting	101
NT : Number of Threads	102
NT : Number of Threads	102
NU : Number of User Queue Elements	103
NU : Number of User Queue Elements	103
NUCID : Adabas Cluster Nucleus ID	104
NUCID : Adabas Cluster Nucleus ID	104
Guidelines	104
OPENRQ : Open Command Required	106
OPENRQ : Open Command Required	106
PLOGDEV : Multiple Protection Log Device Type PLOGSIZE : Multiple Protection Log Size	107
PLOGDEV : Multiple Protection Log Device Type PLOGSIZE : Multiple Protection Log Size	107
PLOGRQ : Protection Log Required	108
PLOGRQ : Protection Log Required	108
PREFETCH : Activate Prefetch Feature	109
PREFETCH : Activate Prefetch Feature	109
PREFICMD : Include Commands in Prefetch/Multifetch Processing	
PREFIFIL : Include Files in Prefetch/Multifetch Processing	111
PREFICMD : Include Commands in Prefetch/Multifetch Processing	
PREFIFIL : Include Files in Prefetch/Multifetch Processing	111

Multiple Inclusions	111
Eligible Commands and Command Pairs	112
PREFNREC : Multifetch Record Count	113
PREFNREC : Multifetch Record Count	113
PREFSBL : Prefetch Buffer Length per Call	114
PREFSBL : Prefetch Buffer Length per Call	114
PREFTBL : Total Size of Prefetch Buffer	115
PREFTBL : Total Size of Prefetch Buffer	115
PREFXCMD (PREFXFIL) : Exclude Commands (Files) from Prefetch/Multifetch Processing .	116
PREFXCMD (PREFXFIL) : Exclude Commands (Files) from Prefetch/Multifetch Processing .	116
Multiple Exclusions	116
Eligible Commands and Command Pairs	117
PROGRAM : Program to Run	118
PROGRAM : Program to Run	118
QBLKSIZE : Block Size for Sequential Datasets	121
QBLKSIZE : Block Size for Sequential Datasets	121
READONLY : Read-Only Session Control	122
READONLY : Read-Only Session Control	122
Guidelines	122
REVIEW : Adabas Review Control	123
REVIEW : Adabas Review Control	123
Dynamic Modification	123
SMGT : Error Handling and Message Buffering Facility	125
SMGT : Error Handling and Message Buffering Facility	125
SORTCACHE : Sort Area Controller Caching	126
SORTCACHE : Sort Area Controller Caching	126
Guidelines	126
SPT : Activate Adabas Triggers and Stored Procedures	127
SPT : Activate Adabas Triggers and Stored Procedures	127
SVC : SVC Number	128
SVC : SVC Number	128
TAPEREL : Tape Handling Control (BS2000 Only)	129
TAPEREL : Tape Handling Control (BS2000 Only)	129
TASKCTGY : Adabas Batch/TP Task Category Control	130
TASKCTGY : Adabas Batch/TP Task Category Control	130
TCPIP : TCP/IP Access Control (OS/390 or z/OS Only)	131
TCPIP : TCP/IP Access Control (OS/390 or z/OS Only)	131
TCPURL : TCP/IP Universal Resource Locator	132
TCPURL : TCP/IP Universal Resource Locator	132
TEMPCACHE : Temp Area Controller Caching	133
TEMPCACHE : Temp Area Controller Caching	133
Guidelines	133
TFLUSH : Synchronous Buffer Flush Duration	134
TFLUSH : Synchronous Buffer Flush Duration	134
TLSCMD : Time Limit for S1, S2, and S4 Complex Searches	135
TLSCMD : Time Limit for S1, S2, and S4 Complex Searches	135
Overriding the Parameter Setting	135
TNAA : Nonactivity Time Limit for Access-Only Users	136
TNAA : Nonactivity Time Limit for Access-Only Users	136
Overriding the Parameter Setting	136

TNAE : Nonactivity Time Limit for ET Logic Users	137
TNAE : Nonactivity Time Limit for ET Logic Users	137
Overriding the Parameter Setting	137
TNAX : Nonactivity Time Limit for Exclusive Control Users	138
TNAX : Nonactivity Time Limit for Exclusive Control Users	138
Overriding the Parameter Setting	138
TT : Transaction Time Limit	139
TT : Transaction Time Limit	139
Overriding the Parameter Setting	139
UEXn : User Exit	140
UEXn : User Exit	140
UTIONLY : Utilities-Only Session Control	141
UTIONLY : Utilities-Only Session Control	141
VISTA : Adabas Vista Control	142
VISTA : Adabas Vista Control	142
WORKCACHE : Work Area Controller Caching	143
WORKCACHE : Work Area Controller Caching	143
Guidelines	143
Operator Commands	144
Operator Commands	144
Entering Operator Commands	144
BS2000 Systems	144
OS/390 or z/OS Systems	145
VM/ESA or z/VM Systems	145
VSE/ESA Systems	145
Operator Commands	146
Nucleus Commands	146
Adabas Utility Operator Command DSTAT	160
SMGT Operator Command	161
SMGT Display Output Samples	165
DISPLAY=ALL	165
DISPLAY=EXITS	165
DISPLAY=MSGBUF	165
DISPLAY=PINS	165
DISPLAY=SUMMARY	166
DISPLAY=LAST	166
Adabas Session Execution	167
Adabas Session Execution	167
Program Synchronization Using Operating System Services	167
BS2000 Session	168
Example of Adabas Session Job Control (BS2000)	168
JCL Required for UES Support (BS2000)	170
OS/390 or z/OS Session	170
Example of Adabas Session Job Control (OS/390 and z/OS)	171
JCL Required for UES Support (OS/390 and z/OS)	171
JCL Required for UES and TCP/IP Support (OS/390 and z/OS)	172
VM/ESA or z/VM Session	172
Example of Adabas Session Job Structure (VM/ESA or z/VM)	173
JCL Required for UES Support (VM/ESA or z/VM)	174
VSE/ESA Session	174
Example of Adabas Session Job Control (VSE/ESA)	174

JCL Required for UES Support (VSE/ESA)	175
Notes	175
Single-User Mode	177
User Profile	177
Profile Table Values	177
Managing the User Profile	178
Adabas Utility Execution	180
Adabas Utility Execution	180
Condition and Termination Codes	180
File and Database Usage	181
Disk Accesses by Utility and Function	183
Possible Resource Conflicts	185
Repetitive Utility Execution Under VM/ESA or z/VM	186
Linking Applications to Adabas	187
Linking Applications to Adabas	187
How the Adabas API Works	187
Online Operation	187
Batch Operation	189
Available Link Routines	190
For IBM Operating Environments	190
For BS2000 Operating Environments	190
Required Work Area	191
Work Area for the Batch/TSO Reentrant ADALNKR	191
Required Application Reentrancy Properties	192
Com-plete : Code Reentrancy Requirements	193
CICS : Code Reentrancy Requirements	193
TSO and IMS/DC (Standalone) : Code Reentrancy Requirements	193
Adabas Control Block Options	193
Using One-Byte File Numbers	194
Using Two-Byte File Numbers	194
Using Both One- and Two-Byte File Numbers in a Single Application	194
Using COBOL to Set the Control Byte	195
Programming Conventions for Issuing Direct Calls	195
Com-plete	195
CICS	196
IMS/DC	198
Shadow II	200
Using the Adabas API in Batch Mode	201
ADAUSER and ADARUN with the Adabas API	201
Batch Execution Modes	202
Support for OpenEdition OS/390 Adabas Clients	204
Option 1 : Link OpenEdition Application with ADALNK	204
Option 2 : Link OpenEdition Application with ADAUSER	205
Setting the OpenEdition Shell Variable STEPLIB	206
Limitations for OpenEdition Support	206
Adabas Restart and Recovery	208
Adabas Restart and Recovery	208
Work Dataset	208
Protection Log	209
Sequential Protection Log	209
Multiple Dataset Protection Log	210

Restart Operations	211
Restart after a User Application Program Failure	211
Restart after an Adabas, Operating System, or Hardware Failure	211
Restart after a Power Failure	212
Using Automatic Restart Management (ARM)	212
Database Recovery	212
Recreating a Database	213
Database or File Recovery Considerations	213
Database Recovery Guidelines	214
Using the Adabas Recovery Aid (ADARAI)	216
Adabas Timeout Feature	217
Adabas Timeout Feature	217
Timeout Parameters	217
Action Taken If Time Limit Exceeded	218
Back Out Transaction	218
Release Command IDs	218
Close User Queue Element	218
Set Timeout Status in the UQE	219

About this Documentation

This documentation contains information needed to operate the Adabas nucleus, Adabas utilities and user programs that call Adabas.

The following summarizes the topics covered by this documentation:

●	Controlling the Adabas Environment	Provides an overview of the Adabas operating structure.
●	Adabas Initialization(ADARUN)	<p>Describes the use of ADARUN control parameters. ADARUN is the Adabas control module and the control statement name. The ADARUN control statement invokes the Adabas nucleus and the Adabas utilities, and defines the Adabas operating environment.</p> <p>User programs are executed according to normal operating system conventions, and communicate with the nucleus in various ways depending on the operating system environment. Batch and TSO user programs normally communicate with Adabas through ADARUN.</p>
●	Operator Commands	<p>Describes Adabas operator commands that can be entered on the console during an Adabas session or during utility operation to terminate an Adabas or user session; display nucleus or utility information; log commands into CLOG; or change Adabas operating parameters or conditions. Commands are listed alphabetically. One command, DSTAT, is listed twice: once as a command for displaying nucleus status, and separately as a command to display current Adabas utility operating status.</p> <p>For Adabas Caching Facility operator commands, see the Adabas Caching Facility documentation. For Adabas Parallel Services operator commands, see the Adabas Parallel Services documentation. For Adabas Cluster Services operator commands, see the Adabas Cluster Services documentation.</p>
●	Adabas Session Execution	Lists the job control statements required to execute a session under the BS2000, OS/390 or z/OS, VM/ESA or z/VM, and VSE/ESA operating systems, and shows examples of the job streams for each system. For examples of utilities jobs, see the Adabas Utilities documentation .
●	Adabas Utility Execution	Describes conditions and requirements for running the Adabas utilities, which are described in detail in the Adabas Utilities documentation .

●	Linking Applications to Adabas	<p>Describes the Adabas link/router APIs and their use, the requirements for single- and multiuser program execution, and the job structure requirements.</p> <p>Since most systems do not allow a standard call to Adabas, Software AG provides an application program interface (API) to translate calls issued by an application program into a form that can be handled by Adabas. Batch applications are supported in both single- and multiuser mode; online operations are controlled by teleprocessing (TP) monitors. The Adabas API is available across all supported mainframe platforms; versions of the API that are specific to particular TP monitors are provided. "Adalink" is a generic term that refers to the portion of the API that is specific to a particular TP monitor.</p>
●	Adabas Restart and Recovery	Provides information and recommended procedures related to restart and recovery.
●	Adabas Timeout Feature	Describes the Adabas time-out feature, a control mechanism for monitoring and restoring unavailable or locked system resources.

Conventions

This document covers the following topics:

- Syntax Conventions
- Dataset Names

Syntax Conventions

The following table describes the conventions used in syntax diagrams of ADARUN parameters and operator commands.

Convention	Description	Example
uppercase, bold	Syntax elements appearing in uppercase and bold font are Adabas keywords. When specified, these keywords must be entered exactly as shown.	<div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;"> ADADBS CHANGE FILE = <i>file-number</i> </div> <p>The syntax elements ADADBS, CHANGE, and FILE are Adabas keywords.</p>
lowercase, italic, normal font	Syntax elements appearing in lowercase and normal, italic font identify items that you must supply.	<div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;"> ADADBS CHANGE FILE = <i>file-number</i> </div> <p>The syntax element <i>file-number</i> identifies and describes the kind of value you must supply. In this instance, you must supply the number of the file affected by the ADADBS CHANGE operation.</p>
mixed case, normal font	Syntax elements appearing in mixed case and normal font (not bold or italic) identify items established by other Adabas control statements. This notation is usually used to identify how default values are determined for some parameters in Adabas syntax.	<div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;"> [SORTDEV = { <i>device-type</i> ADARUN-device }] </div> <p>The syntax element "ADARUN-device" indicates that the device type identified by the ADARUN DEVICE parameter will be used if a different device type is not specified. The literal "ADARUN-device" should <i>not</i> be specified for the SORTDEV parameter.</p>

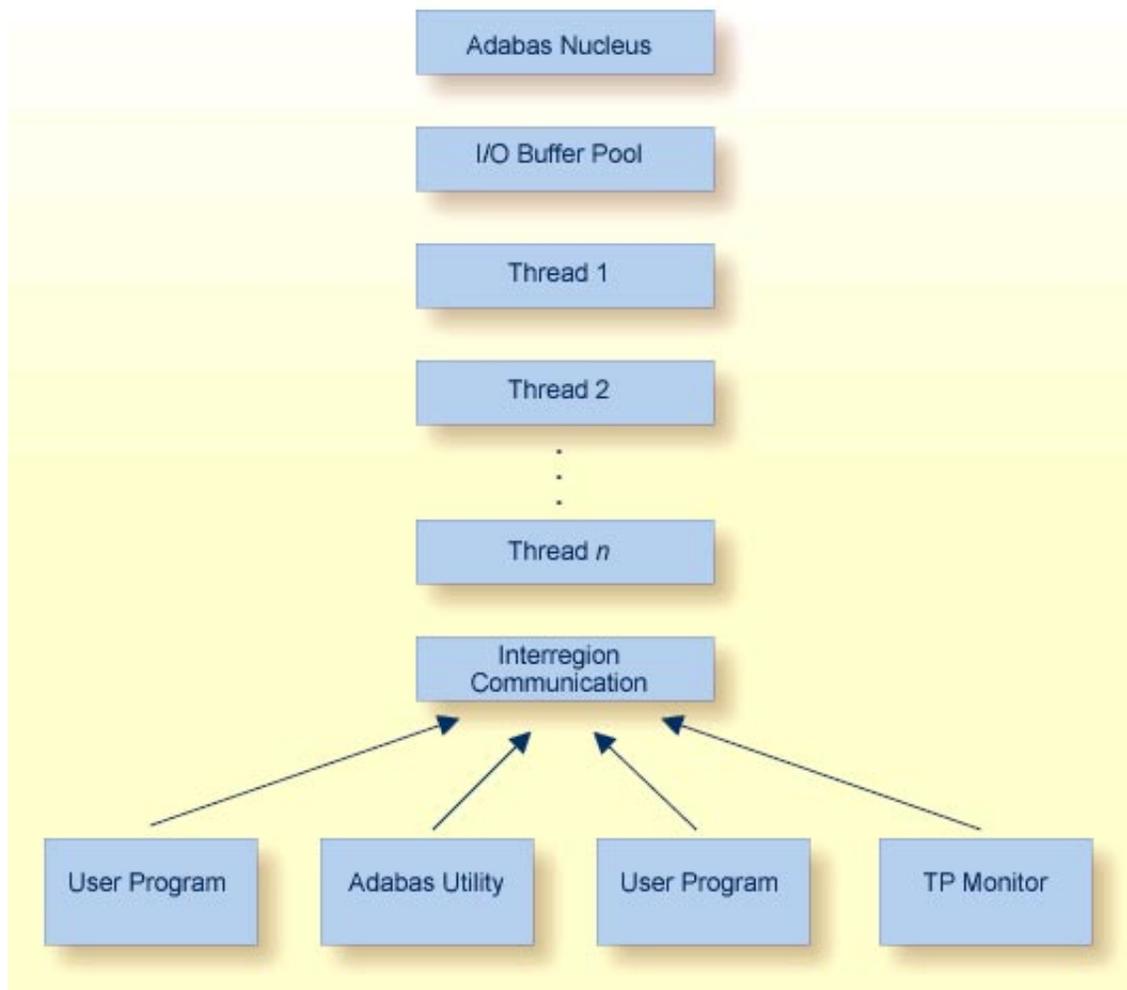
Convention	Description	Example
underlining	<p>Underlining is used for two purposes:</p> <ol style="list-style-type: none"> 1. To identify default values, wherever appropriate. Otherwise, the defaults are explained in the accompanying parameter descriptions. 2. To identify the short form of a keyword. 	<div data-bbox="735 212 1156 289" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p><u>[LRECL = {record-buffer-length 4000 }]</u></p> </div> <p>In the example above, 4000 is the default that will be used for the LRECL parameter if no other record buffer length is specified.</p> <div data-bbox="735 464 859 541" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p><u>DEVICE</u></p> </div> <p>In the example above, the short version of the DEVICE parameter is DE.</p>
vertical bars ()	<p>Vertical bars are used to separate mutually exclusive choices.</p> <p>Note: In more complex syntax involving the use of large brackets or braces, mutually exclusive choices are stacked instead.</p>	<div data-bbox="735 674 1156 751" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p><u>ADAORD { RESTRUCTUREF REF }</u></p> </div> <p>In the example above, you must select RESTRUCTUREF or REF for this ADAORD function. There are no defaults.</p>
brackets ([])	<p>Brackets are used to identify optional elements. When multiple elements are stacked or separated by vertical bars within brackets, only one of the elements may be supplied.</p>	<div data-bbox="735 1035 1156 1140" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p><u>SORTSEQ = { descriptor [,MU] [,NU] ISN [, STARTISN = value] physical-sequence }</u></p> </div> <p>In this example, the SORTSEQ parameter and the MU, NU, and STARTISN subparameters are optional.</p> <p>Note: Note that the mutually exclusive choices for the SORTSEQ parameter are stacked.</p>
braces ({ })	<p>Braces are used to identify required elements. When multiple elements are stacked or separated by vertical bars within brackets, one and only one of the elements must be supplied.</p>	<div data-bbox="735 1396 1284 1522" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p><u>SUSPEND [TTSYN = {time-available-to-sync ADARUN-TT }] [TRESUME = {time-until-resume 120 }] RESUME</u></p> </div> <p>In this example, either the SUSPEND or RESUME parameter is required.</p>
indentation	<p>Indentation is used to identify subparameters of a parameter.</p>	<div data-bbox="735 1661 1284 1787" style="border: 1px solid black; padding: 5px; margin-bottom: 10px;"> <p><u>SUSPEND [TTSYN = {time-available-to-sync ADARUN-TT }] [TRESUME = {time-until-resume 120 }] RESUME</u></p> </div> <p>In this example, TTSYN and TRESUME are subparameters of the SUSPEND parameter.</p>

Convention	Description	Example
ellipsis (...)	Ellipses are used to identify elements that can be repeated. If other punctuation is included before a ellipsis, the punctuation must be used to separate repeated elements.	<div data-bbox="735 212 1117 289" style="border: 1px solid black; padding: 2px; margin-bottom: 10px;">[FIELD = 'field-name [, option]... '] ...</div> <p>In this example, the FIELD parameter can be repeated. In addition, the options associated with the field can be repeated.</p>
other punctuation and symbols	All other punctuation and symbols must be entered exactly as shown.	<div data-bbox="735 447 1117 525" style="border: 1px solid black; padding: 2px; margin-bottom: 10px;">[FIELD = 'field-name [, option]... '] ...</div> <p>In this example, the single quotation marks must be specified around the field definitions and their associated options. In addition, options must be separated by commas.</p>

Dataset Names

Dataset names starting with DD are referred to in Adabas manuals with a slash separating the DD from the remainder of the dataset name to accommodate VSE/ESA dataset names that do not contain the DD prefix. The slash is not part of the dataset name.

Controlling the Adabas Environment



Adabas Operating Structure

The Adabas *nucleus* manages all database activities; for example, data access/update.

The Adabas *utilities* are used to perform database service functions; for example, loading/deleting files.

A *user program* calls the Adabas nucleus in order to perform database access/update.

For controlling the Adabas environment, the database administrator can use either the optional Adabas Online System or the ADADBS utility.

The Adabas nucleus and each user program or Adabas utility are executed in a separate storage area defined by the operating system:

Operating System	Storage Area Name
BS2000	task
OS/390 or z/OS	address space / data space / hiperspace
VM/ESA or z/VM	virtual machine
VSE/ESA	partition / address space / data space

For consistency and simplification, this manual refers to all BS2000, OS/390 or z/OS, and VSE/ESA areas (task, address space, partition, region, etc.) as *regions*. VM/ESA or z/VM areas are called *virtual machines*.

Adabas Initialization (ADARUN Statement)

The ADARUN control statement defines and starts the Adabas operating environment. The ADARUN control statement also starts Adabas utilities.

ADARUN:

- loads the ADAIOR module, which performs database I/O and other functions that depend on the operating system;
- interprets the ADARUN parameter statements, then loads and modifies the appropriate Adabas nucleus or utility modules according to the ADARUN parameter settings; and
- transfers control to Adabas.

The ADARUN statement, normally a series of entries each specifying one or more ADARUN parameter settings, is specified in the DDCARD (OS/390, z/OS, VM/ESA, z/VM, or BS2000) or VSE/ESA CARD dataset. For more specific job information, refer to the appropriate location in this documentation.

Note:

The ADARUN parameters described in this chapter apply to the standard Adabas configuration and to Adabas cluster configurations. Other ADARUN parameters must also be specified for such features as: Adabas Caching Facility, Adabas Delta Save Facility, Adabas Transaction Manager. Refer to the related documentation for more information about the ADARUN parameters that may be required.

The ADARUN documentation is organized under the following headings:

- ADARUN Parameter Directory
- ADARUN Parameter Syntax
- Guidelines for Setting Adabas Session Parameters

The ADARUN parameter directory provides a tabular overview of ADARUN parameters.

ADARUN Parameter Directory

Parameter	Usage
<u>AOSLOG</u>	Log to DDPRINT commands issued by AOS or ADADBS OPERCOM that modify the active nucleus
<u>AREXCLUDE</u>	Exclude file(s) from autorestart
<u>ARMNAME</u>	Automatic restart management (ARM) program name
<u>ASSOCACHE</u>	Controller caching control for the Associator component (OS/390, z/OS, VM/ESA, and z/VM only)
<u>ASYTVS</u>	Asynchronous buffer flush based on vol-ser
<u>CACHE</u>	Adabas Caching Facility control
<u>CDXnn</u>	Collation descriptor user exit(s)
<u>CLOGDEV</u>	Device for multiple command log datasets
<u>CLOGLAYOUT</u>	Define command log format
<u>CLOGMRG</u>	Merge cluster command logs automatically
<u>CLOGSIZE</u>	Size (blocks) for multiple command log datasets
<u>CLUCACHENAME</u>	Adabas cluster cache structure/area name
<u>CLUCACHESIZE</u>	Adabas cluster global cache area size
<u>CLUGROUPNAME</u>	Adabas cluster group name
<u>CLULOCKNAME</u>	Adabas cluster lock structure/area name
<u>CLULOCKSIZE</u>	Adabas cluster global lock area size
<u>CLUSTER</u>	Adabas cluster nucleus session control
<u>CMADDR</u>	Start address for common memory pool (BS2000 only)
<u>CMDQMODE</u>	Command queue memory pool location (BS2000 only)
<u>CMFIX</u>	Fix location for common memory pool (BS2000 only)
<u>CMLADDR</u>	See CMADDR
<u>CMLFIX</u>	See CMFIX
<u>CMLSCOPE</u>	See CMSCOPE
<u>CMLSIZE</u>	See CMSIZE
<u>CMSCOPE</u>	Access to common memory pool (BS2000 only)
<u>CMSIZE</u>	Common memory pool use and size (BS2000 only)
<u>CT</u>	Command time limit (seconds)
<u>DATACACHE</u>	Controller caching control for the Data Storage component (OS/390, z/OS, VM/ESA, and z/VM only)
<u>DBID</u>	Database ID (physical)

Parameter	Usage
<u>DEVICE</u>	Device type
<u>DIRRATIO</u> / <u>ELEMENTRATIO</u>	Together define the ratio of directory entries to data elements in Adabas cluster environments
DSF	Delta Save Facility control
<u>DSFEX1</u>	Delta Save Facility user exit
<u>DTP</u>	Distributed transaction processing control
DUALCLD	Dual command log device
DUALCLS	Dual command log size (blocks)
DUALPLD	Dual protection log device
DUALPLS	Dual protection log size (blocks)
<u>FASTPATH</u>	Adabas Fastpath control
<u>FMXIO</u>	Limit parallel I/O operations by LFIOP flush processing
<u>FORCE</u>	Overwrite database ID
<u>GROUPS</u>	BS2000 interprocess scope
HEXnn	Hyperdescriptor exit(s)
<u>IDTNAME</u>	ID table name (BS2000 only)
<u>IGNDIB</u>	Ignore DIB entry
<u>IGNDTP</u>	Ignore distributed transaction processing area (Work part 4)
<u>INTNAS</u>	Interval between nucleus statistic checkpoints (SYNS 60)
<u>LBP</u>	Length of buffer pool
<u>LCP</u>	Length of security pool
<u>LDEUQP</u>	Length of unique (UQ) descriptor pool
<u>LDTP</u>	Length of distributed transaction processing area (Work part 4)
<u>LFIOP</u>	Length of asynchronous flush pool
LFP	Length of internal format buffer pool
LI	Length of ISN list table (TBI)
<u>LOCAL</u>	Nucleus for local use only; unknown to network
LOGCB	Log control block
<u>LOGCLEX</u>	Log command log extension (CLEX)
<u>LOGFB</u>	Log format buffer
<u>LOGGING</u>	Log Adabas commands
LOGIB	Log ISN buffer
LOGIO	Log I/O activity

Parameter	Usage
<u>LOGRB</u>	Log record buffer
LOGSB	Log search buffer
<u>LOGSIZE</u>	Maximum command log size
<u>LOGUX</u>	Log user exit B data
<u>LOGVB</u>	Log value buffer
LP	Length of data protection area (Work part 1)
LQ	Length of sequential command table
<u>LRDP</u>	Length of the cluster block update "redo" pool
LS	Length of sort area
LU	Length of intermediate user buffer
<u>LWKP2</u>	Length of ISN list processing area (Work part 2)
LWP	Length of Adabas work pool
<u>MODE</u>	Mode of operation
<u>MSGBUF</u>	Size of the message buffer
<u>MSGCONSL</u>	Display console messages in uppercase
<u>MSGDRUCK</u>	Display messages sent to DD/DRUCK in uppercase
<u>MSGPRINT</u>	Display messages sent to DD/PRINT in uppercase
<u>MXMSG</u>	Maximum communication time limit between Adabas sysplex cluster nuclei
<u>MXTNA</u>	Maximum inactivity time limit override for a user
<u>MXTSX</u>	Maximum Sx execution time limit override for a user
MXTT	Maximum transaction time limit override for a user
<u>NAB</u>	Number of attached buffers
NC	Number of command queue elements
<u>NCLOG</u>	Number of command log datasets
NH	Number of hold queue elements
<u>NISNHQ</u>	Number of ISNs in hold queue for user
<u>NONDES</u>	Non-descriptor searches
<u>NPLOG</u>	Number of protection log datasets
<u>NQCID</u>	Number of active command IDs per user
<u>NSISN</u>	Number of ISNs per ISN table element
NT	Number of threads
NU	Number of user queue elements
<u>NUCID</u>	Adabas cluster nucleus ID

Parameter	Usage
<u>OPENRQ</u>	Open command required
<u>PLOGDEV</u>	Device for multiple protection log datasets
<u>PLOGRQ</u>	Protection log required
<u>PLOGSIZE</u>	Size (blocks) for multiple protection log datasets
<u>PREFETCH</u>	Prefetch/multifetch feature control
<u>PREFICMD</u>	Include command in prefetch/multifetch
<u>PREFIFIL</u>	Include file in prefetch/multifetch
<u>PREFNREC</u>	Multifetch record count
<u>PREFSBL</u>	Prefetch single buffer length
<u>PREFTBL</u>	Prefetch total buffer length
<u>PREFXCMD</u>	Exclude command from prefetch/multifetch
<u>PREFXFIL</u>	Exclude file from prefetch/multifetch
<u>PROGRAM</u>	Program to run
<u>QBLKSIZE</u>	Sequential dataset block size (optimized by ADAIOR)
<u>READONLY</u>	Read-only session control
<u>REVIEW</u>	Adabas Review control
<u>SMGT</u>	Error handling (PIN) facility control
<u>SORTCACHE</u>	Controller caching control for the Adabas sort area component (OS/390, z/OS, VM/ESA, and z/VM only)
<u>SPT</u>	Adabas triggers and stored procedures control
<u>SVC</u>	SVC number
<u>TAPEREL</u>	Tape processing control (BS2000 only)
<u>TASKCTGY</u>	Task category control (BS2000 only)
<u>TCPIP</u>	TCP/IP access control (OS/390 and z/OS only)
<u>TCPURL</u>	TCP/IP universal resource locator (URL)
<u>TEMPCACHE</u>	Controller caching for the Adabas temp area component (OS/390, z/OS, VM/ESA, and z/VM only)
<u>TFLUSH</u>	Synchronous buffer flush time
<u>TLSCMD</u>	Time limit for S1, S2, and S4 complex searches (seconds)
TNAA	Non-activity time limit (access-only users)
TNAE	Non-activity time limit (ET logic users)
TNAX	Non-activity time limit (exclusive update users)
TT	Transaction time limit
UEXn	User exits

Parameter	Usage
<u>U</u> TIONLY	Utilities-only session
<u>V</u> ISTA	Adabas Vista control
<u>W</u> ORKCACHE	Controller caching for the Adabas work area component (OS/390, z/OS, VM/ESA, and z/VM only)

ADARUN Parameter Syntax

The syntax for the ADARUN statement and parameters is:

```
ADARUN parameter = value,...
```

where *parameter=value* is one of the ADARUN parameters described in this section. Multiple *parameter=value* entries separated by commas may be included in a statement. The comma following the last *parameter=value* entry of a statement is ignored.

All *parameter=value* entries must end before position 73. Any *parameter=value* entries that would extend beyond position 72 must be coded on a new statement. Positions 73-80 are ignored.

The literal "ADARUN" must be entered in positions 1-6 of each ADARUN statement. Any number of blanks is permitted between "ADARUN" and the first parameter, but no blanks are permitted within the *parameter=value* string. A blank following a *parameter=value* entry indicates the end of the statement.

An asterisk (*) in position 1 indicates a user comment line.

The first ADARUN statement cannot continue beyond position 72. The second statement represents a continuation of the first statement. All ADARUN continuation statements have the same format and restrictions as the first statement.

The ADARUN parameters are summarized in the following table, and described in detail in the sections following the table.

Unless noted otherwise, each parameter has a default value that ADARUN uses if the parameter is not specified.

Some parameter names can be abbreviated. The minimum acceptable abbreviation for a given parameter is shown by the underlined part of the parameter name. Parameter names shown without underlining cannot be abbreviated.

Setting Adabas Session Parameters

This chapter is a user's guide to setting Adabas session parameters on the ADARUN control statement. This information is organized as follows:

- General Guidelines
 - Using Session Statistics to Determine Parameter Settings
 - Parameters for Operating Systems
-

General Guidelines

Use the following guidelines when setting the various Adabas session parameters:

1. Ensure that the following are specified correctly:
 - program to be executed (see PROGRAM parameter).
 - mode of operation (MODE parameter).
 - database ID (see DBID parameter).
2. If the session is to be restricted to read only (no updating permitted), use the READONLY parameter.
3. If the session is to be restricted to Adabas utilities only, use the UTIONLY parameter.
4. Review buffer and table sizes to be in effect for the session:

Parameter	Description
ASYTVS, LFIOP, TFLUSH	Buffer flush control
LBP	Length of the buffer pool
LCP	Length of security pool
LDEUQP	Length of unique (UQ) descriptor pool
LDTP	Length of distributed transaction processing area (Work part 4) for use with DTP=RM
LFP	Length of internal format buffer pool
LI	Length of table for ISN lists
LP	Length of data protection area (Work part 1)
LQ	Length of table of sequential commands
LRDP	Length of the cluster block update "redo" pool
LS	Length of sort area
LU	Length of intermediate user buffer
LWKP2	Length of Work part 2
LWP	Length of Adabas work pool
MSGBUF	Length of message buffer for use with SMGT=YES
NAB	Number of attached buffers
NC	Number of elements in command queue
NH	Number of elements in hold queue
NISNHQ	Maximum number of records in hold queue per user
NQCID	Maximum number of active command IDs per user
NSISN	Maximum number of ISNs per TBI element
NT	Number of threads
NU	Number of elements in user queue

5. Review the time limits to be in effect for the session:

Parameter	Description
CT	Command timeout limit
MXTNA	Maximum inactivity limit using OP command for individual user
MXTSX	Maximum execution limit for Sx command for individual user
MXTT	Maximum transaction time limit using OP command for individual user
TLSCMD	Sx command time limit
TNAA	Inactivity limit for access only users
TNAE	Inactivity limit for ET logic users
TNAX	Inactivity limit for EXU users
TT	Transaction time limit

6. If dual protection logging is to be used, the DUALPLD and DUALPLS parameters apply.
7. If multiple (2-8) protection log datasets are to be used, the NPLOG, PLOGDEV, and PLOGSIZE parameters apply.
8. If command logging is to be used, the following parameters are applicable:

CLOGLAYOUT
 LOGGING
 LOGCB, LOGCLEX, LOGFB, LOGIB, LOGIO, LOGRB, LOGSB, LOGVB

9. If a sequential command log is to be used, the LOGGING parameter applies.
10. If dual command log datasets are to be used, the DUALCLD and DUALCLS parameters apply.
11. If multiple (2-8) command log datasets are to be used, the NCLOG, CLOGDEV, and CLOGSIZE parameters apply.
12. If the multifetch or prefetch feature is to be used, the following parameters are applicable:

PREFETCH
 PREFICMD
 PREFIFIL
 PREFNREC
 PREFSBL
 PREFTBL
 PREFXCMD
 PREFXFIL

13. Determine if any of the following user exits are to be activated:

User Exit	Description
CDXnn	Collation (sort) descriptor processing
DSFEX1	Delta Save Facility user exit for use with DSF=YES
HEXnn	User hyperdescriptor processing
UEX1	User processing before command execution
UEX2	Dual log copy
UEX3	User phonetization
UEX4	User processing after command processing and before command logging
UEX5	Adabas Review hub event handler
UEX6	User processing before data compression
UEX8	Message/utility user processing
UEX9	User processing during file unload
UEX12	Multiple log dataset copy

14. Enable the nucleus to run with specific subsystems and associated products or in certain environments:

Parameter	Description
CACHE	Adabas Caching Facility (ACF)
CLUSTER	Adabas cluster nucleus session control (ALS or ASM)
DSF	Adabas Delta Save Facility (ADE)
DTP	Enable the nucleus for distributed transaction processing as a resource or transaction manager (ATM)
FASTPATH	Adabas Fastpath (AFP)
REVIEW	Adabas Review (REV)
SMGT	Enhanced error handling (PINs) and message buffering subsystem
STP	Triggers and Stored Procedures Facility
VISTA	Adabas Vista (AVI)

15. If CLUSTER is either LOCAL or SYSPLEX, determine the cluster environment settings:

Parameter	Description
CLOGMRG	Merge cluster command logs automatically
CLUCACHENAME	Custer cache structure/area name
CLUCACHESIZE	Parallel cluster global cache area size
CLUGROUPNAME	Cluster name
CLULOCKNAME	Custer lock structure/area name
CLULOCKSIZE	Parallel cluster global lock area size
DIRRATIO/ELEMENTRATIO	Together define the ratio of directory entries to data elements
LRDP	Length of the cluster block update "redo" pool
MXMSG	Maximum communication time limit between sysplex cluster nuclei
NUCID	Cluster nucleus ID

16. Determine the status of the nucleus in a network:

Parameter	Description
LOCAL	Enable nucleus for network connections
TCPIP	TCP/IP direct link to the nucleus
TCPURL	Identify the location of the TCP/IP link

17. Determine if any of the following parameters are applicable for the session:

Parameter	Description
DEVICE	Device type for first block of Associator
FORCE	Ignore active database ID setting in the ID table
IGNDIB	Ignore active nucleus entry in data integrity block (DIB)
IGNDTP	Ignore data in Work part 4 on startup for use with DTP=RM
NONDES	Non-descriptor searches
OPENRQ	Open command required
PLOGRQ	Selective/normal protection log required
QBLKSIZE	Sequential block size

18. When performing an autorestart following an ABEND, use the optional AREXCLUDE parameter to exclude any problem files from processing.

Using Session Statistics to Determine Parameter Settings

The Adabas session statistics are printed automatically at the end of a session and may be displayed during the session. These statistics can help you determine the optimum settings for each parameter. Adabas Online System may also be used to obtain session statistics.

Parameters for Operating Systems

Some ADARUN parameters are operating-system dependent, and must be set according to the particular operating environment.

BS2000

Parameter	Description
CMADDR	Start address for common memory pool
CMDQMODE	Command queue memory pool location
CMFIX	Fix location for common memory pool
CMLADDR	Below 16MB; see CMADDR
CMLFIX	Below 16MB; see CMFIX
CMLSCOPE	Below 16MB; see CMSCOPE
CMLSIZE	Below 16MB; see CMSIZE
CMSCOPE	Access to common memory pool
CMSIZE	Use and size of common memory pool
GROUPS	Resource naming restrictions
IDTNAME	ID table name
TAPEREL	Tape handling
TASKCTGY	Adabas task category control

OS/390, z/OS, and VSE/ESA or Compatible Systems

Parameter	Description
SVC	OS/390, z/OS, or VSE/ESA SVC assignment for Adabas

OS/390, z/OS, VM/ESA, and z/VM Systems Only

Parameter	Description
xxxxCACHE	Disable controller caching for Adabas components, as desired. Caching is active if not explicitly disabled.

AOSLOG : Log Update Commands from AOS and ADADBS

Parameter	Specify . . .	Possible Values	Default
<u>AOSLOG</u>	whether to log update commands from AOS or ADADBS that modify the active nucleus	YES NO	NO

Value	Meaning
YES	Log to DD/PRINT Adabas calls that modify the active nucleus (not read/display calls) issued by ADADBS OPERCOM or Adabas Online System (AOS).
NO	The default setting. Commands are not logged.

Example:

```
ADARUN PROG=ADANUC,AOSLOG=YES
```

AREXCLUDE : Exclude Files from Autorestart Processing

Parameter	Specify . . .	Possible Values	Default
<u>AREXCLUDE</u>	the file(s) to exclude from autorestart processing.	1-5000	none

The AREXCLUDE parameter is intended for use only when autorestart processing fails (ABENDs) due to response codes on one or more Adabas files. In this case, excluding the problem file(s) from autorestart might allow for successful autorestart processing on the other files.

Excluded files are marked as "utility use only". Software AG recommends that you restore excluded files from the latest save tape and regenerate them.

Multiple file numbers can be specified on the same parameter if separated by commas.

Example:

Exclude files 199 and 231 from autorestart processing.

```
ADARUN PROG=ADANUC,AREXCLUDE=199,231
```

ARMNAME : Activate ARM

Parameter	Specify . . .	Possible Values	Default
ARMNAME	and activate a particular automatic restart management (ARM) program.	arm-pgm-name	none

Automatic restart management (ARM) is used to automatically restart a nucleus when it ABENDs. It is suppressed when the ABEND is intentional; for example, when it results from a parameter error.

ARM can be used in both cluster and noncluster nucleus environments.

ARM programs ('elements') are defined by the system programmer in the ARM 'policy'. They specify when, where, and how often a restart is to be attempted.

The ARMNAME parameter has no effect if an ARM policy has not been defined.

The name can be 1-16 characters long. In a cluster environment, the name must be unique across the cluster. Uppercase alphabetic characters, the numbers 0-9, and the special characters \$, @, and #, and the underscore (_) may be used. The first character may not be numeric. Names that start with A through I and with SYS are reserved for use by IBM.

Example:

Activate the automatic restart management program RESTART002.

```
ADARUN PROG=ADANUC,ARMNAME=RESTART002
```

ASSOCACHE : Associator Controller Caching

This parameter applies to the operating systems OS/390, z/OS, VM/ESA, and z/VM only.

Parameter	Specify . . .	Possible Values	Default
ASSOCACHE	whether to enable (YES) or disable (NO) controller caching for the Associator (ASSO) dataset.	YES NO	YES

Value	Meaning
YES	The default setting. ASSO blocks are pre-read and maintained as long as possible in the caching facilities of the disk controller.
NO	ASSO blocks are not pre-read.

Notes:

1. If ASSO blocks are highly scattered or the update frequency is low, controller caching can cause longer I/O response times.
2. ADARUN caching control parameters also exist for Data Storage (DATACACHE), the work area (WORKCACHE), the sort area (SORTCACHE), and the temp area (TEMPCACHE).

Example:

Run the nucleus without controller caching for the ASSO dataset.

```
ADARUN PROG=ADANUC,ASSOCACHE=NO
```

ASYTVS : Asynchronous Bufferflush

Parameter	Specify . . .	Possible Values	Default
<u>ASYTVS</u>	whether to flush buffers asynchronously based on volume serial number.	YES NO	YES

Example:

Disable asynchronous buffer flush based on volume.

```
ADARUN PROG=ADANUC,ASYTVS=NO
```

CACHE : Adabas Caching Facility Control

Parameter	Specify . . .	Possible Values	Default
<u>CACHE</u>	whether ADACSH is to be loaded during Adabas session initialization	YES NO	NO

Example:

ADACSH is loaded during the Adabas session initialization. ADACSH operator commands are accepted and dynamic caching is in effect.

```
ADARUN CACHE=YES
```

CDXnn : Collation Descriptor User Exit

Parameter	Specify . . .	Possible Values	Default
CDXnn	the collation descriptor exit and the user routine to be used with it.	see text	none

Used in conjunction with the universal encoding subsystem (UES), the format of the collation descriptor user exit parameter is

CDXnn=exit-name

where

nn is the number of the collation descriptor exit, a two-digit decimal integer in the range 01-08 inclusive.

exit-name is the name of the user routine that gets control at the collation descriptor exit; the name can be up to 8 characters long.

Only one program may be specified for each collation descriptor exit. Up to 8 collation descriptor exits may be specified (in any order). See the User Exits documentation for more information.

Example:

Run the Adabas nucleus using collation descriptor exits 1 and 2. Give control to the user routine "UCOLDE1" at collation descriptor exit 1; give control to the user routine "UCOLDE2" at collation descriptor exit 2.

ADARUN PROG=ADANUC,CDX01=UCOLDE1,CDX02=UCOLDE2

CLOGDEV : Multiple Command Log Device Type

CLOGSIZE : Multiple Command Log Size

Parameter	Specify ...	Possible Values	Default
CLOGDEV	the device type to be used for multiple command logs.	see text	value of DEVICE parameter
CLOGSIZE	the number of blocks available for each command log.	Min 16, Max 16 777 215	none

The CLOGDEV and CLOGSIZE parameters specify the type of external storage device used for multiple command logs and the number of blocks available for each command log dataset. These parameters must be used *together*; if you specify one parameter, you must also specify the other parameter.

You must also specify the NCLOG parameter to use multiple command logs. If you specify a user exit, it must be UEX12.

The Adabas Installation documentation specifies the device types that Adabas supports.

All command logging datasets must be formatted using the ADAFRM utility before they are used.

The maximum size of the dataset depends on the limits of the operating system and on the requirements of the user.

Example:

Run the Adabas nucleus with command logging. Use 6 command log datasets. Use a model 3380 device for command logging. The size of each command log dataset is 1350 blocks, based on an allocation of 10 cylinders. (Each cylinder of a 3380 device has 15 tracks, and each track can contain 9 blocks of command log data. Thus, 10 cylinders = 10 15 9 = 1350 blocks).

```
ADARUN PROG=ADANUC,LOGGING=YES,NCLOG=6,CLOGDEV=3380,CLOGSIZE=1350
```

CLOGLAYOUT : Command Logging Format

Parameter	Specify . . .	Possible Values	Default
<u>CLOGLAYOUT</u>	the command log format.	4 5	5

This parameter specifies the format of the Adabas command log.

Value	Meaning
4	Use the command log format from Adabas 4 and 5.1. A program designed to analyze the old command log format should specify CLOGLAYOUT=4.
5	The default setting. Use the command log format from Adabas 5.2 and above. User programs designed for Adabas 5.2 or later should specify CLOGLAYOUT=5.

Example:

Use the default format of the command log.

```
ADARUN PROG=ADANUC,CLOGL=5
```

CLOGMRG : Cluster Command Log Merge

Parameter	Specify . . .	Possible Values	Default
CLOGMRG	whether the Adabas cluster is to run with automatic CLOG merge.	YES NO	NO

An automatic CLOG merge process provides a single cluster-wide command log containing, in chronological order, all Adabas commands executed by any of the cluster nuclei in the time period covered by the log.

CLOGMRG specifies whether the Adabas cluster is to run with the automatic CLOG merge:

Value	Action
YES	enables an automated process that copies dual or multiple command logs to sequential datasets and merges those datasets across a cluster into a single cluster-wide command log. This process automatically merges command logs when an ADARES CLCOPY is submitted from user exit 2 and executed. The cluster nuclei and the ADARES utility perform coordinated CLOG switches and CLCOPY functions with merging of CLOG records, much the same way the PLOG merge process works.
NO	(the default), each nucleus produces its own CLOG, independent of any other nucleus in the cluster; dual or multiple CLOGs can be manually copied using ADARES CLCOPY to a sequential dataset; and the resulting sequential CLOGs can be manually merged across a cluster using the ADARES MERGE CLOG function.

CLOGMRG is a global (that is, a cluster-wide) parameter, which means that the setting of this parameter in the first cluster nucleus to become active is propagated to all nuclei that subsequently become active.

Parameter Dependencies

Whatever the setting of CLOGMRG, a nucleus writes CLOG records only according to its own LOGGING and LOGxx parameters.

CLOGMRG is effective only in an Adabas cluster environment; that is, when CLUSTER is specified with a value other than NO and NUCID is specified with a nonzero value.

Dynamic Modification

The setting of the CLOGMRG parameter can be changed dynamically using the CLOGMRG command from the operator console, the ADADBS OPERCOM CLOGMRG function, or the Modify Parameter function of Adabas Online System.

Because CLOGMRG is a global parameter, the change request to one nucleus is automatically propagated to all nuclei in the cluster.

CLUCACHENAME : Cluster Cache Structure/Area Name

Parameter	Specify . . .	Possible Values	Default
CLUCACHENAME	the name of the cache structure/area of the Adabas cluster in which the current nucleus participates.	text	none

For Adabas Cluster Services, the CLUCACHENAME identifies the particular sysplex cache structure (defined in the CFRM policy) associated with a particular Adabas sysplex nucleus cluster. The same name must be set in each nucleus in the cluster. CLUCACHENAME must be unique for each cluster; that is, each database.

Adabas Parallel Services automatically creates a default name for each global cache area allocated by ADACOM. However, if you wish to control the name of the global cache area, you can set this parameter.

Note:

Software AG recommends that you allow the system to build the default name.

For Adabas Parallel Services, the CLUCACHENAME identifies a particular global cache area associated with all cluster nuclei that have the same SVC/DBID set as the current nucleus. The same name must be set in each cluster nucleus that has the same SVC/DBID set. CLUCACHENAME must be unique for each SVC/DBID set.

In all cases, the name can be 1-16 characters long. Valid characters are numeric, uppercase alphabetic, the special characters \$, @, and #. The name must begin with an alphabetic character (A-Z). IBM names begin with SYS, UNDESIG, or the characters A through I (these characters are not reserved).

Note:

Additionally, the special character underscore (_) is allowed in Adabas Cluster Services; that is, sysplex environments.

Example:

```
ADARUN PROG=ADANUC, CLUCACHENAME=SAG1SMP#CACHE1
```

CLUCACHESIZE : Size of the Global Cache Area

Parameter	Specify . . .	Minimum	Maximum	Default
CLUCACHESIZE	the amount of storage to allocate for the global cache area that services the Adabas Parallel Services cluster in which the current nucleus participates.	128K	2G	none

Note:

This parameter is effective only when CLUSTER=LOCAL and NUCID is a nonzero value.

The CLUCACHESIZE parameter specifies the amount of storage (that is, the size of the data space (memory pool for BS2000)) to be allocated for the global cache area that will service all nuclei in the Adabas Parallel Services cluster in which the current nucleus participates. ADACOM allocates all storage space above the 16-megabyte line.

If no value is specified, the global lock area is not allocated; there is no default.

Each Adabas Parallel Services nucleus has its own local buffer pool sized by the ADARUN parameter LBP. In addition, Adabas Parallel Services uses a global cache area to ensure that current data is available to all Adabas Parallel Services cluster nuclei.

The global cache area must be large enough to retain

- "directory elements" for all RABNs that reside in all the buffer pools; and
- enough "data elements" to keep changed RABNs between buffer flushes (cast-outs).

The ADARUN parameters DIRRATIO and ELEMENTRATIO determine the ratio between the number of directory and data elements.

Size may be specified in bytes, in kilobytes followed by a "K", in megabytes followed by an "M", or in gigabytes followed by a "G":

- the minimum size is 128000 bytes (or 128K);
- the maximum size is 2047475456 bytes (or 2047144K; or 2047M; or 2G); and
- there is no default for the global cache area. If a size is not specified, no cache area is allocated.

The value is always rounded up to the nearest 4-kilobyte boundary.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session. The "buffer efficiency" statistic can be used to determine the optimal size of the parameter.

Example:

ADARUN PROG=ADANUC,CLUCACHESIZE=

CLUGROUPNAME : Cluster Group Name

Parameter	Specify ...	Possible Values	Default
CLUGROUPNAME	the name of the XCF messaging group in Adabas Cluster Services or the cluster in Adabas Parallel Services	text	none

The CLUGROUPNAME parameter provides

- the XCF group name to be used by the sysplex cluster nuclei for their communication using XCF.
- the name to be used by the Adabas Parallel Services cluster nuclei for their communication.

All nuclei in the same cluster must have the same CLUGROUPNAME. Nuclei belonging to different clusters must have different CLUGROUPNAMEs.

In addition, the CLUGROUPNAME must be different from the XCF group names used by other subsystems that communicate using XCF.

The name can be 1-8 characters long. Valid characters are numeric, uppercase alphabetic, and the special characters \$, @, and #. The name must begin with an alphabetic character (A-Z). IBM names begin with SYS, UNDESIG, or the characters A through I (these characters are not reserved).

Example:

```
ADARUN  PROG=ADANUC,CLUGROUPNAME=SAG1CLU
```

CLULOCKNAME : Cluster Lock Structure/Area Name

Parameter	Specify . . .	Possible Values	Default
CLULOCKNAME	the name of the lock structure defined in the coupling facility's CFRM policy or the global lock area in which the current nucleus participates	text	none

For Adabas Cluster Services, CLULOCKNAM identifies a particular sysplex lock structure (defined in the CFRM policy) associated with a particular Adabas nucleus cluster. The same name must be set in each nucleus in the cluster. CLULOCKNAME must be unique for each cluster; that is, each database.

Adabas Parallel Services automatically creates a default name for each global lock area allocated by ADACOM. However, if you wish to control the name of the global lock area, you can set this parameter.

Note:

Software AG recommends that you allow the system to build the default name.

For Adabas Parallel Services, the CLULOCKNAME identifies a particular global lock area associated with all Adabas Parallel Services cluster nuclei that have the same SVC/DBID set as the current nucleus. The same name must be set in each cluster nucleus that has the same SVC/DBID set. CLULOCKNAME must be unique for each SVC/DBID set.

In all cases, the name can be 1-16 characters long. Valid characters are numeric, uppercase alphabetic, the special characters \$, @, and #. The name must begin with an alphabetic character (A-Z). IBM names begin with SYS, UNDESIG, or the characters A through I (these characters are not reserved).

Note:

Additionally, the special character underscore (_) is allowed in Adabas Cluster Services; that is, sysplex environments.

Example:

```
ADARUN PROG=ADANUC,CLULOCKNAME=SAG1SMP#LOCK1
```

CLULOCKSIZE : Size of the Global Lock Area

Parameter	Specify ...	Minimum	Maximum	Default
CLULOCKSIZE	the size of the global lock area	128K	2G	none

Note:

This parameter is effective only when CLUSTER=LOCAL and NUCID is a nonzero value.

The CLULOCKSIZE parameter specifies the amount of storage (that is, the size of the data space (memory pool for BS2000)) to be allocated for the Adabas Parallel Services global lock area that will service the Adabas Parallel Services cluster in which the current nucleus participates. ADACOM allocates all storage space above the 16-megabyte line.

If no value is specified, the global lock area is not allocated; there is no default.

The size of a lock entry is 128 bytes.

In addition, you need to plan space for lock names that are longer than 40 bytes and unique descriptor values that are longer than 32 bytes. For such unique descriptor value locks, a lock name element is allocated in the global lock area that is equal in length to the unique descriptor value. The value for the ADARUN LDEUQP parameter provides an upper limit for this estimate.

The global lock area size can be estimated using the formula

$$\begin{aligned} & \text{RND2 (RND2 (MAXFILES) * 4 + RND2 ((LDEUQP / 16) * 2) } \\ & \quad + \text{RND2 (NH * 2) } \\ & \quad + \text{RND2 (NU * 2) * 2) * 4 + (MAXFILES * 4 + LDEUQP / 16 + NH + NU * 2) * 128} \end{aligned}$$

where 'RND2' means rounded up to the next power of 2. 'MAXFILES' is the maximum number of files in the database set in ADADEF or ADAORD. 'LDEUQP', 'NH', and 'NU' are Adabas ADARUN parameters.

Size may be specified in bytes, in kilobytes followed by a "K", in megabytes followed by an "M", or in gigabytes followed by a "G":

- the minimum size is 128000 bytes (or 128K);
- the maximum size is 2047475456 bytes (or 2047144K; or 2047M; or 2G); and
- there is no default for the global cache area. If a size is not specified, no cache area is allocated.

The value is always rounded up to the nearest 4-kilobyte boundary.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

Example:

With the nucleus parameters NH=4000, NU=200, LDEUQP=20000, and the maximum number of files in the database (MAXFILES) as 255:

```
RND2 (RND2 (255) * 4 + RND2 ( (20000 / 16) * 2) + RND2 (4000 * 2)
+ RND2 (200 * 2) * 2) * 4 + (255 * 4 + 20000 / 16 + 4000 + 200 * 2) * 128=
RND2 (1024 + 4096 + 8192 + 1024) * 4 + 6670 *
128=16384 * 4 + 853760=919296=circa 1 MB
```

To this value, add the LDEUQP parameter value (20000) for eventual unique descriptor values to arrive at the maximum size required for the global lock area:

```
ADARUN PROG=ADANUC,CLULOCKSIZE=1M
```

CLUSTER : Adabas Nucleus Cluster Session Control

Parameter	Specify . . .	Possible Values	Default
<u>CLUSTER</u>	whether or not (NO) to enable Adabas Cluster Services (SYSPLEX) or Adabas Parallel Services (LOCAL) for the session.	NO SYSPLEX LOCAL	NO

This parameter is used to specify the type of cluster a nucleus will participate in for the current session. If this parameter is set to SYSPLEX or LOCAL

- a NUCID must also be specified for the nucleus.
- MODE=MULTI must be specified. See the ADARUN parameter MODE
- READONLY=NO is automatically set by the system.

Example:

Include the nucleus in and Adabas sysplex cluster:

```
ADARUN CLUSTER=SYSPLEX,NUCID=1, ...
```

CMADDR and CMLADDR : GETMAIN Memory Pool Start Address

These parameters apply to the BS2000 operating system only.

Parameter	Specify . . .	Minimum	Maximum	Default
	the start address of the GETMAIN memory pool			
CMADDR	above the 16MB line	0, or X'1000000'	X'FE00000'	0
CMLADDR	below the 16MB line	0, or X'10000'	X'E00000'	0

Note:

These parameters should only be used if required by an installation site, in which case the site will provide the needed value.

These parameters are involved with setting all required memory into a common memory pool for use by associated tasks on BS2000 above (CMADDR) or below (CMLADDR) the 16 megabyte line. They set the start address of this memory pool.

If the default value (0) is used for an ..ADDR parameter, the start address depends on the value of the corresponding ..SIZE parameter:

- If the ..ADDR parameter value is 0 and the corresponding ..SIZE parameter is *not* 0, the operating system will set the start address at the next available megabyte boundary.
- If the ..ADDR parameter value is 0 and the corresponding ..SIZE parameter is *also* 0, required memory will not be obtained in a common memory pool but rather in class 6 memory.

Example:

Allow for three (3) megabytes of user storage in the address range X'2000000' to X'4FFFFFF':

```
ADARUN PROG=ADANUC,CMADDR=2000000,CMSIZE=2500000
```

Allow for three (3) megabytes of user storage below 16 megabytes in the address range X'200000 to X'4FFFFFF:

```
ADARUN PROG=ADANUC,CMLADDR=200000,CMLSIZ=2500000
```

CMDQMODE : Command Queue Mode

This parameter applies to the BS2000 operating system only.

Parameter	Specify . . .	Possible Values	Default
CMDQMODE	whether to allocate the command queue memory pool below or above the 16-MB line.	BELOW ABOVE	BELOW

CMDQMODE specifies whether to allocate the BS2000 memory pool for the Adabas command queue below or above the 16-MB line.

Value	Meaning
BELOW	The default setting. Places the BS2000 memory pool for the Adabas command queue below the 16-MB line in one or more 64-kilobyte segments.
ABOVE	Places the BS2000 memory pool for the Adabas command queue above the 16-MB line in one or more 1-MB segments.

Example:

Place the Adabas command queue memory pool above the 16-MB line in 1-MB segments.

```
ADARUN PROG=ADANUC,CMDQMODE=ABOVE
```

CMFIX or CMLFIX : GETMAIN Memory Pool Fixed Location

These parameters apply to the BS2000 operating system only.

Parameter	Specify . . .	Possible Values	Default
	whether the GETMAIN memory pool must have a fixed location or not		
<u>CM</u> FIX	above the 16MB line	YES NO	NO
<u>CM</u> LFIX	below the 16MB line	YES NO	NO

Note:

Use these parameters only if required by an installation site, in which case the site will provide the needed value.

The ..FIX parameter is ignored if the corresponding ..SIZE parameter value is 0.

These parameters are involved with setting all required memory into a common memory pool for use by associated tasks on BS2000 above (CMFIX) or below (CMLFIX) the 16 megabyte line. This parameter specifies whether the memory pool is at a fixed location or not:

Value	The application's GETMAIN memory pool . . .
NO	need not be located at the same start address in the subtask as in the main task.
YES	must be located at the same start address in the subtask as in the main task.

Note:

A subtask depends on the application. An example of this is Adabas Review in local mode.

Example:

Allow for three (3) megabytes of user storage above 16 megabytes in the address range X'2000000' to X'4FFFFFF'. This address range will also be used by the subtask:

```
ADARUN PROG=ADANUC,CMADDR=2000000,CMSIZE=2500000,CMFIX=YES
```

Allow for three (3) megabytes of user storage below 16 megabytes in the address range X'200000' to X'4FFFFFF'. This address range will also be used by the subtask:

```
ADARUN PROG=ADANUC,CMLADDR=200000,CMSIZE=2500000,CMFIX=YES
```

CMLxxxxx Parameters

The discussion of these parameters is paired with related parameters as follows:

CMLADDR
CMLFIX
CMLSCOPE
CMLSIZE

CMSCOPE or CMLSCOPE : GETMAIN Memory Pool Scope

These parameters apply to the BS2000 operating system only.

Parameter	Specify . . .	Possible Values	Default
	access to the GETMAIN memory pool		
<u>C</u> MSCOPE	above the 16MB line	GROUP GLOBAL	GROUP
<u>C</u> MSCOPE	below the 16MB line	GROUP GLOBAL	GROUP

Note:

These parameters should only be used if required by an installation site, in which case the site will provide the needed value.

The ..SCOPE parameter is ignored if the corresponding ..SIZE parameter value is 0.

These parameters are involved with setting all required memory into a common memory pool for use by associated tasks on BS2000 above (CMSCOPE) or below (CMLSCOPE) the 16 megabyte line. This parameter specifies accessibility to the memory pool:

Value Meaning

GROUP The GETMAIN common memory pool can only be accessed by other tasks using the same user ID.

GLOBAL The GETMAIN common memory pool can only be accessed by all user IDs in the system.

Example:

Allow for three (3) megabytes of user storage at the next available megabyte boundary above the 16MB line accessible to tasks with the same user ID.

```
ADARUN PROG=ADANUC ,CMSIZE=2500000 ,CMSCOPE=GROUP
```

Allow for three (3) megabytes of user storage at the next available megabyte boundary below the 16MB line accessible to tasks with the same user ID.

```
ADARUN PROG=ADANUC ,CMLSIZE=2500000 ,CMLSCOPE=GROUP
```

CMSIZE or CMLSIZE : GETMAIN Memory Pool Use and Size

These parameters apply to the BS2000 operating system only.

Parameter	Specify . . .	Minimum	Maximum	Default
	whether the GETMAIN is performed in a (decimal) common memory pool and if so, its size in bytes			
<u>CMSIZE</u>	above the 16MB line	0	address-limit	0
<u>CMLSIZE</u>	below the 16MB line	0	address-limit	0

Note:

These parameters should only be used if required by an installation site, in which case the site will provide the needed value.

These parameters are involved with setting all required memory into a common memory pool for use by associated tasks on BS2000 above (CMSIZE) or below (CMLSIZE) the 16 megabyte line. These parameters determine whether a common memory pool is used and if so, its size:

- If the ..SIZE parameter value is 0, the default, the required application memory is obtained in the class 6 memory of the system.
- If the ..SIZE parameter value is *not* 0, the required application memory is obtained in a common memory pool of the specified size rounded up to the next megabyte.

Example:

Allow for three (3) megabytes of user storage at the next available megabyte boundary above the 16 megabyte line accessible to tasks with the same user ID.

```
ADARUN PROG=ADANUC,CMSIZE=2500000
```

Allow for three (3) megabytes of user storage at the next available megabyte boundary below the 16 megabyte line accessible to tasks with the same user ID.

```
ADARUN PROG=ADANUC,CMLSIZE=2500000
```

CT : Command Timeout Limit

Parameter	Specify . . .	Minimum	Maximum	Default
CT	the maximum time for interregion communication of results from Adabas to the user.	1	16 777 215	60

The maximum number of seconds (more precisely, units of 1.048576 seconds) that can elapse from the time an Adabas command has been completed until the results are returned to the user through the interregion communication (operating-system-dependent).

This parameter is used to prevent a command queue element and attached buffer from being held for a long period of time for a user who has terminated abnormally.

Possible causes of a command timeout are

- user region is swapped out or cannot be dispatched;
- user is cancelled;
- user has low priority in high activity system.

If the CT limit is exceeded,

- a BT command is executed (if applicable and necessary);
- the command queue element and attached buffer are released;
- a message ADAM93 is printed; and
- if the user has not terminated, response code 254 is returned to the user program.

Note:

This parameter should not be confused with the nonactivity (TNAA, TNAE, TNAX) and transaction time limit (TT) parameters, which are based on user activity.

Example:

Permit about 30 seconds to obtain a result through interregion communication.

```
ADARUN PROG=ADANUC,CT=30
```

DATACACHE : Data Storage Controller Caching

This parameter applies to the operating systems OS/390, z/OS, VM/ESA, and z/VM only.

Parameter	Specify . . .	Possible Values	Default
DATACACHE	whether to enable (YES) or disable (NO) controller caching for Data Storage (DATA) dataset.	YES NO	YES

Value Meaning

YES The default setting. Data Storage blocks are pre-read and maintained as long as possible in the caching facilities of the disk controller.

NO Data Storage blocks are not pre-read.

Notes:

1. If Data Storage blocks are highly scattered or the update frequency is low, controller caching can cause longer I/O response times.
2. ADARUN caching control parameters also exist for the Associator (ASSOCACHE), the work area (WORKCACHE), the sort area (SORTCACHE), and the temp area (TEMPCACHE).

Example:

Run the nucleus without controller caching for the Data Storage dataset.

```
ADARUN PROG=ADANUC,DATACACHE=NO
```

DBID : Database Identification

Parameter	Specify . . .	Minimum	Maximum	Default
<u>DBID</u>	the physical ID of the database.	1	65 535	1

Example:

Execute the ADAINV utility against physical database 2.

```
ADARUN PROG=ADAINV,DBID=2
```

DEVICE : Device Type

Parameter	Specify . . .	Possible Values	Default
<u>DEVICE</u>	the device type on which the first block of the Associator is stored.	see text	2000 (BS2000) 3380 (other)

The device type of the external storage device on which the first block of the Associator is stored.

This device type is also used as default for most device type parameters.

The Adabas Installation documentation specifies the device types supported by Adabas.

Some device type designations have been reserved for special use such as VSAM dataset support on OS/390 or z/OS systems. The following is a partial list of reserved device types and their uses:

Device	Use
5555	VSAM DD/xxxxR5
6666	VSAM DD/xxxxR4
7777	VSAM DD/xxxxR3
8888	VSAM DD/xxxxR2
9999	VSAM DD/xxxxR1 - (default for self-defined VSAM block sizes)

For more information about VSAM device usage, see *Device and File Considerations* in the Adabas Installation documentation.

Example:

Execute the ADALOD utility; store the first block of the Associator on a 3390 device.

```
ADARUN PROG=ADALOD,DEVICE=3390
```

DIRRATIO / ELEMENTRATIO : Cluster Cache Directory to Elements Ratio

The DIRRATIO and ELEMENTRATIO parameters together define the ratio of directory entries to data elements in the Adabas Cluster Services coupling facility cache structure or the Adabas Parallel Services global cache area.

This ratio influences the allocation of the cache structure/area when the first cluster nucleus starts. It has no influence on the cache structure/area when a subsequent nucleus starts.

Parameter	Specify . . .	Minimum	Maximum	Default
<u>DIRRATIO</u>	the directory portion of the directory-to-element ratio	1	32767	4
<u>ELEMENTRATIO</u>	the element portion of the directory-to-element ratio	1	32767	1

The DIRRATIO / ELEMENTRATIO ratio is optionally used in Adabas cluster environments to tune the cache structure/area for better performance.

Directory elements are used to keep track of the cluster member that has a particular RABN in the cache so that the RABN can be invalidated should another member modify it.

If the number of directory elements is insufficient, existing directory elements are reused and the RABNs associated with those directory elements are invalidated and can then no longer be tracked. When RABNs become invalid even though they have not been modified, they must be reread and registered again the next time they are referenced and validated.

It is generally better to reassign storage for data elements to keep more ASSO and DATA blocks in the cache structure/area than to define too many directory elements in the cache. More data elements than necessary can be used to keep additional RABNs to improve the cache efficiency.

The number of directory elements need not be greater than the sum of the sizes of all buffer pools divided by the smallest block size in use for ASSO and DATA.

Change the default value only when the coupling facility or system monitor reports that cache directory entries are being reclaimed or that entry space is being wasted. Such reports are available using the system activity monitor available with your operating system.

When connecting to the cache structure/area during startup, the ADAX57 message reports the number of directory and data elements allocated and the size of a data element.

Example:

```
ADAX57  dbid CONNECTED TO CACHE STRUCTURE cache-name
ADAX57  dbid DIRECTORY ELEMENTS           19,312
ADAX57  dbid DATA ELEMENTS                4,828
ADAX57  dbid DATA ELEMENT SIZE           1,024
```

DSF : Adabas Delta Save Facility Control

Parameter	Specify . . .	Possible Values	Default
DSF	whether to enable support for the (optional) Delta Save Facility.	YES NO	NO

The DSF parameter controls the Delta Save Facility for an Adabas nucleus or utility. If the Delta Save Facility is not installed on your system, specifying this parameter may cause an error.

Value Meaning

YES Loads certain Delta Save Facility program modules and runs the nucleus or utility (as specified in the ADARUN PROG parameter) in DSF mode. Delta Save functions are enabled: the DLOG area can be installed or removed, Delta Save operations can be performed, full and delta save tapes can be merged. DSF=YES is required for Delta Save functions of the nucleus and the ADASAV, ADARES, and ADAULD utilities; it is optional for other functions and utilities. It is required for ADASAV SAVE database (if a DLOG area exists), SAVE DELTA, MERGE, and RESTORE DELTA; for ADARES PLCOPY (only if online saves are performed), for COPY (only for rebuilding the DSIM dataset); and for ADAULD UNLOAD SAVETAPE, if the inputs are full and delta save tapes produced by DSF.

NO (default) Runs the nucleus or utility in non-DSF mode. If DSF=NO is specified or the parameter is omitted, Delta Save functions cannot be performed. Specifying DSF=NO after specifying DSF=YES switches the nucleus from DSF to non-DSF mode. If a DLOG area is defined when the nucleus is started and DSF=NO is specified, the nucleus removes the DLOG area. If a DLOG area is defined when the nucleus is started, the DSF parameter may not be omitted. You can manually remove the DSF logging area from the nucleus using the "Remove DSF" command through Adabas Online System.

Example:

Run the Adabas nucleus in DSF mode; performing Delta Save functions is possible.

```
ADARUN PROG=ADANUC,DSF=YES
```

DSFEX1 : Delta Save Facility Exit

Parameter	Specify . . .	Possible Values	Default
<u>DSFEX1</u>	the user routine to be given control by the Delta Save Facility.	see text	none

Used in conjunction with the Delta Save Facility, the format of the user exit parameter is

DSFEX1=exit-name

where

`exit-name` is the name of the user routine that gets control at the DSF user exit; the name can be up to 8 characters long.

Only one program may be specified for the DSF user exit. Only one DSF user exit may be specified. See the Adabas Delta Save Facility documentation for more information.

Example:

Use the Delta Save Facility with the user exit routine DSAVEPGM.

ADARUN PROG=ADANUC,DSF=YES,DSFEX1=DSAVEPGM

DTP : Distributed Transaction Processing Control

Parameter	Specify . . .	Possible Values	Default
<u>DTP</u>	whether to enable distributed transaction processing and, if so, whether the nucleus is to be a resource or transaction manager.	NO RM TM	NO

The DTP parameter controls the distributed transaction processing status of the Adabas nucleus. If Adabas Transaction Manager is not installed on a separate nucleus in your system, specifying this parameter may cause an error.

Value	Meaning
NO	(default) The nucleus does not support distributed transaction processing.
RM	<p>The nucleus is to participate in distributed transaction processing as a resource manager (RM) with other resource managers (databases) coordinated by a transaction manager. This setting is only possible when</p> <ul style="list-style-type: none"> - MODE=MULTI - READONLY=NO <p>If DTP=RM, the LDTP parameter specifies the size of the Work part 4 area that the nucleus uses to keep information about distributed transactions.</p>
TM	<p>The nucleus supports distributed transaction processing as a transaction manager (TM). Only one such nucleus is allowed per operating system instance. This setting is only possible when</p> <ul style="list-style-type: none"> - MODE=MULTI - the nucleus is not part of a cluster (Parallel Services or Cluster Services) - LOCAL=NO - READONLY=NO

Example:

The nucleus will participate as a resource manager in a distributed transaction processing system coordinated by a transaction manager.

```
ADARUN PROG=ADANUC,DTP=RM
```

DUALCLD : Dual Command Log Device Type

DUALCLS : Dual Command Log Size

Parameter	Specify . . .	Possible Values	Default
DUALCLD	the device type to be used for dual command logging.	see text	value of DEVICE parameter
DUALCLS	the number of blocks available for each dual command log.	Min 16, Max 16 777 215	none

The DUALCLD and DUALCLS parameters specify the type of external storage device used for dual command logging and the number of blocks available for each dual command log dataset. These parameters must be used *together*; if you specify one parameter, you must also specify the other parameter. If you specify a user exit, it must be UEX2.

The Adabas Installation documentation specifies the device types that Adabas supports.

Both dual command logging datasets must be formatted using the ADAFRM utility before they are used.

The maximum size of the dataset depends on the limits of the operating system and on the requirements of the user.

Example:

Run the Adabas nucleus with command logging. Use a model 3380 device for dual command logging. The size of each command log dataset is 1350 blocks, based on an allocation of 10 cylinders. (Each cylinder of a 3380 device has 15 tracks, and each track can contain 9 blocks of command log data. Thus, 10 cylinders = 10 15 9 = 1350 blocks).

```
ADARUN PROG=ADANUC,LOGGING=YES,DUALCLD=3380,DUALCLS=1350
```

DUALPLD : Dual Protection Log Device Type

DUALPLS : Dual Protection Log Size

Parameter	Specify . . .	Possible Values	Default
DUALPLD	the device type to be used for dual protection logging.	see text	value of DEVICE parameter
DUALPLS	the number of blocks available for each dual protection log.	Min 16, Max 16 777 215	none

The DUALPLD and DUALPLS parameters specify the type of external storage device used for dual data protection logging and the number of blocks available for each log dataset. These parameters must be used *together*; if you specify one parameter, you must also specify the other parameter. If you specify a user exit, it must be UEX2.

When these parameters are specified, dual protection logging is activated for the session. See *Adabas Restart and Recovery* for information about dual protection logging.

The dual protection log device type must be the same as that of the Adabas Work dataset. The Adabas Installation documentation specifies the device types that Adabas supports.

Both dual protection log datasets must be formatted using the ADAFRM utility before they are used. The appropriate JCL for the datasets is required.

Example:

Run the Adabas nucleus with dual data protection logging. Use a model 3380 device; the size of each dual log dataset is 1200 blocks, based on an allocation of 10 cylinders. (Each cylinder of a 3380 device has 15 tracks, and each track contains 8 blocks of protection log data. Thus, 10 cylinders = 10 * 15 * 8 = 1200 blocks).

```
ADARUN  PROG=ADANUC ,DUALPLD=3380 ,DUALPLS=1200
```

FASTPATH : Adabas Fastpath Control

Parameter	Specify . . .	Possible Values	Default
<u>F</u> ASTPATH	whether to enable Adabas Fastpath support.	YES NO	NO

The FASTPATH parameter controls the optional Adabas Fastpath product. If Adabas Fastpath is installed on your system, FASTPATH=YES enables the full functionality of the product.

If Adabas Fastpath is not installed on your system, FASTPATH=YES invokes the Adabas Fastpath command analysis sampler AFPLOOK that is provided with Adabas.

The demo version of Adabas Online System that is supplied with Adabas provides access to the Adabas Fastpath online system SYSAFP so that the AFPLOOK sample can be viewed online exactly as it would appear if Adabas Fastpath were installed.

Example:

Run Adabas Fastpath with Adabas.

```
ADARUN PROG=ADANUC,FASTPATH=YES
```

FMXIO : Parallel LFIOP I/O Operations

Parameter	Specify . . .	ASYTVS=	Minimum	Maximum	Default
FMXIO	the maximum number of parallel I/O operations by LFIOP flush processing.	YES NO	1 1	16 100	1 60

The LFIOP parameter enables asynchronous buffer flush operation and sets the I/O pool size. The FMXIO parameter sets the limit on the number of I/O operations that can be started in parallel by LFIOP flush processing.

Parameter Dependencies

The precise meaning of the FMXIO parameter depends on the setting of the ASYTVS parameter):

- When ASYTVS=YES (perform buffer flushes by volume), FMXIO specifies the number of I/Os to be started in parallel *on each volume*. The minimum and default number is 1; the maximum number is 16. If the specified number exceeds that, it is automatically reduced to 16.
- When ASYTVS=NO (perform buffer flushes in ascending RABN sequence, without regard to the distribution of the blocks over volumes), FMXIO specifies the number of I/Os to be started in parallel overall. The minimum number is 1; the maximum value is 100; and the default is 60.

Example:

A maximum of 30 I/O operations for asynchronous buffer flushes can be started in parallel:

```
ADARUN PROG=ADANUC,ASYTVS=NO,FMXIO=30
```

Dynamic Modification

The setting of FMXIO can be modified dynamically using the command FMXIO=nnn from the operator console or the Modify Parameter function of Adabas Online System.

FORCE : Allow Database ID Table Entry Overwrite

Parameter	Specify . . .	Possible Values	Default
<u>FORCE</u>	whether the nucleus can overwrite an existing ID table entry.	YES NO	NO

When an Adabas nucleus starts up, ADARUN scans the ID table to ensure that no entry exists for the nucleus. To protect the database from concurrent updates, the nucleus is not allowed to access the database if an ID table entry exists for the nucleus.

The ID table entry is derived from the database ID and the job name. The ID table entry is deleted when the nucleus terminates normally.

The FORCE parameter allows the nucleus to overwrite the existing ID table entry and access the database.



Warning:

Do not use the FORCE parameter unless absolutely necessary, or the integrity of the database could be lost. Ensure that no nucleus is active for the ID table entry being overwritten.

Note:

In an Adabas Cluster Services or Adabas Parallel Services environment, the FORCE parameter applies to the NUCID, rather than the database ID, because a cluster nucleus builds an ID table entry for the NUCID.

Value	Meaning
YES	The nucleus that is starting can overwrite an existing ID table entry. FORCE=YES is required when restarting a session that terminated abnormally with an ADAM98 message. In this case, the ID table still contains an active entry for the nucleus. Overwriting the existing entry by specifying FORCE=YES prevents further communication to the overwritten nucleus and causes loss of cross-memory environment resources, which cannot be restored until the next IPL.
NO	(default) If the ID table contains an entry for the nucleus that is starting, the nucleus is denied access to the database.

Example:

If the ID table contains an active entry for DBID 7, overwrite the entry.

```
ADARUN PROG=ADANUC, FORCE=YES, DBID=7
```

GROUPS : User Group Interprocess Communication

This parameter applies to the BS2000 operating system only.

Parameter	Specify . . .	Possible Values	Default
<u>G</u> ROUPS	whether BS2000 interprocess communication is limited to users with the same logon ID.	YES NO	NO

This parameter limits the accessibility of ADARER and its ID table (IDT) to a group of users with the same logon ID.

Value	Meaning
YES	Allows the user(s) access to the IDT that starts under the same logon ID as the Adabas nucleus. When initialized, the IDT is set to GROUPS=YES; all users (nuclei, utilities, Entire Net-work, and all user tasks) must therefore also specify GROUPS=YES in either the ADARUN statement or using the Adalink parameters.
NO	The default setting. All users on one machine have access to this IDT, even if they are logged on under a different logon ID.

Note:

More than one IDT with the same name can be used if one is set to GROUPS=NO (the default) and the rest specify GROUPS=YES.

Example:

```
ADARUN PROG=ADANUC, DBID=11, GROUPS=YES
```

HEXnn :Hyperdescriptor User Exit

Parameter	Specify . . .	Possible Values	Default
HEXnn	the hyperdescriptor exit and the user routine to be used with it.	see text	none

Used in conjunction with the hyperdescriptor facility, the format of the parameter is

HEXnn=exit-name

where

nn is the number of the hyperdescriptor exit, a two-digit decimal integer in the range 01-31 inclusive.

exit-name is the name of the user routine that gets control at the hyperdescriptor exit; the name can be up to 8 characters long.

Only one program may be specified for each hyperdescriptor exit. Up to 31 hyperdescriptor exits may be specified (in any order). See the User Exits documentation for more information.

Example:

Run the Adabas nucleus using hyperdescriptor exits 1 and 2. Give control to the user routine "UHYPER1" at hyperdescriptor exit 1; give control to the user routine "UHYPER2" at hyperdescriptor exit 2.

ADARUN PROG=ADANUC,HEX01=UHYPER1,HEX02=UHYPER2

IDTNAME : Define ID Table Name

This parameter applies to the BS2000 operating system only.

Parameter	Specify . . .	Possible Values	Default
<u>IDTNAME</u>	an alternate ID table.	ADAccccc	ADABAS5B

Defines a name for an (alternate) Adabas environment. The name must be eight characters long, and begin with "ADA...".

Note:

All users of the new ID table must include the IDTNAME in either the ADARUN statement or Adalink parameters.

Example:

Define the name "ADAOURDB" for the second Adabas environment.

```
ADARUN PROG=ADANUC, IDTN=ADAOURDB
```

IGNDIB : Ignore DIB Entry for Adabas Nucleus

Parameter	Specify . . .	Possible Values	Default
<u>IGNDIB</u>	whether to ignore and delete an active nucleus entry in the data integrity block.	YES NO	NO

The "DIB" is the data *integrity* block in the Associator. The DIB contains an entry that indicates whether an Adabas session is active for the database. This feature prevents an update nucleus session from being started while another one is running on the same database; it is especially important when several CPUs share the same database DASD devices. The DIB entry is automatically released when the Adabas session is terminated normally.

This parameter indicates whether this update nucleus session should ignore and delete the entry in the data integrity block (DIB) in the Associator.

Value	Meaning
YES	Adabas ignores and deletes the existing DIB entry and initiates a new session.
NO	The default setting. If the DIB entry indicates that an Adabas session is already active for the database, Adabas rejects an attempt to start an update session against the database. This is true even when the session that the DIB indicates is active has terminated abnormally.

Note:

Software AG recommends that you use `IGNDIB=YES` only once and immediately remove it from the JCL. If no DIB entry exists when `IGNDIB=YES` is specified, the nucleus terminates abnormally.

Example:

Execute an Adabas update session; ignore and delete the DIB entry for an active Adabas update nucleus.

```
ADARUN PROG=ADANUC,IGNDIB=YES
```

IGNDTP : Ignore Two-Phase Commit Area (Work Part 4)

Parameter	Specify . . .	Possible Values	Default
<u>IGNDTP</u>	whether to ignore the two-phase commit area (Work part 4).	YES NO	NO



Warning:

This parameter is for emergency use only; contact your Software AG technical support representative before using it. Whenever the information on Work part 4 is ignored, the integrity of the incomplete global transactions that are related to that information cannot be guaranteed.

IGNDTP=YES can be specified with DTP=NO or DTP=RM.

When DTP=NO, the nucleus will not start if there is data in Work part 4 related to the recovery of previously incomplete global transactions. In this situation, specifying IGNDTP=YES forces the nucleus to ignore the data in Work part 4 and start anyway.

When DTP=RM, the nucleus also ignores any data on Work part 4 and reinitializes the area. This can be useful during a recovery when the state and correctness of the incomplete global transactions that are stored on Work part 4 has already been established.

Example:

Execute an Adabas update session without support for distributed transactions; ignore data in the Work part 4 area left over from the previous session with support for distributed transaction processing.

```
ADARUN PROGRAM=ADANUC ,DTP=NO ,LDTP=0 ,IGNDTP=YES
```

INTNAS : Time Interval Between SYNS 60 Checkpoints

Parameter	Specify . . .	Minimum	Maximum	Default
<u>I</u> NTNAS	the time interval between SYNS 60 checkpoints.	1	16777215	3600

The SYNS 60 statistic checkpoint is generated by the Adabas nucleus and written to the Adabas Checkpoint system file. It may be displayed using the CPLIST function of the ADAREP utility or Adabas Online System. For more information about checkpoints see the ADAREP utility discussion in the Adabas Utilities documentation.

INTNAS allows you to set the time interval (in units of 1.048576 seconds) between SYNS 60 checkpoints. The default (3600) is approximately 1 hour meaning that nucleus statistics will be recorded once each hour.

Example:

Set Adabas so that nucleus statistics are recorded every 30 minutes.

```
ADARUN PROGRAM=ADANUC,INTNAS=1800
```

LBP : Length of Buffer Pool

Parameter	Specify . . .	Minimum	Maximum	Default
<u>LBP</u>	the maximum size of the Adabas buffer pool.	80 000	2147483647	350000

The maximum number of bytes to be used for the Adabas buffer pool during a session.

Adabas allocates the number of bytes specified to the Adabas buffer pool if sufficient memory is available. A minimum of 80000 bytes must be available to successfully initiate a session. For performance reasons, however, Software AG recommends that you specify at least 50000 bytes per thread. Ensure that sufficient memory is available to accommodate the LBP size specified. If asynchronous buffer flush management is used (that is, the LFIOP parameter specifies a nonzero value), the LBP value must be greater than that specified by LFIOP.

In environments running in 31-bit addressing mode, the buffer pool space is allocated above the 16-megabyte line.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

The "buffer efficiency" statistic can be used to determine the optimal size of the parameter.

Example:

Allocate a maximum of 800,000 bytes to the Adabas buffer pool for this session.

```
ADARUN PROG=ADANUC,LBP=800000
```

LCP : Length of Security Pool

Parameter	Specify . . .	Minimum	Maximum	Default
<u>L</u> CP	the maximum size of the Adabas security pool.	2000	16 777 215	10 000

The maximum number of bytes to be used for the Adabas security pool during a session. The pool is used to keep security information in memory in order to optimize performance.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

Example:

Allocate a maximum of 50,000 bytes to the Adabas security pool for this session.

```
ADARUN PROG=ADANUC,LCP=50000
```

LDEUQP : Length of Unique Descriptor Pool

Parameter	Specify . . .	Minimum	Maximum	Default
<u>L</u> DEUQP	the size of the unique descriptor pool.	0	16 777 215	5000

The unique (UQ) descriptor pool comprises elements that are the length of a unique descriptor value plus 11 bytes. The UQ descriptor pool should be large enough to hold all unique descriptors that are being changed by *all* transactions at any point in time.

In environments running in 31-bit addressing mode, the LDEUQP pool is allocated above the 16-megabyte line.

Example:

Assuming that the average length of unique descriptor values is about 9 bytes, this Adabas session can process a maximum of about 50 unique descriptors at one time.

```
ADARUN PROG=ADANUC,LDEUQP=1000
```

LDTP : Length of Distributed Transaction Processing Area (Work Part 4)

Parameter	Specify . . .	Minimum	Maximum	Default
<u>LDTP</u>	the number of blocks to allocate to Work part 4.	0	65535	0

This parameter specifies the length of the distributed transaction processing area, Work part 4.

If DTP=RM and LDTP is either unspecified or specified as '0', the length of Work part 4 is equivalent to that specified for Work part 1 in the ADARUN LP parameter.

If specified, the ADARUN LDTP must be equal to or greater than the LP parameter value (Work part 1). If you specify a value for LDTP that is less than the value for LP, Adabas changes the specified value to equal the LP value.

When Work part 4 contains data about incomplete transactions, it may be increased in size but not decreased. Work part 4 must be empty before it can be made smaller.

Example:

The LP parameter specifies 2000 blocks for the data protection area (Work part 1). Use 2500 blocks for the distributed transaction processing area on the Adabas Work dataset (part 4).

```
ADARUN PROG=ADANUC,DTP=RM,LP=2000,LDTP=2500
```

LFIOP : Asynchronous Buffer Pool Control

Parameter	Specify . . .	Minimum	Maximum	Default
<u>LFIOP</u>	whether to enable asynchronous buffer flush operation and set the I/O pool size.	80000	2147483647	0

The asynchronous buffer flush I/O pool enables asynchronous Adabas buffer flush operation and defines the size of the related buffer. Specifying a non-zero value for LFIOP enables asynchronous buffer flush operation; the value specified determines the buffer size, in bytes.

If LFIOP is not specified or zero is specified (the default), synchronous buffer flushing is performed (as in Adabas releases before version 5.2). Note that flushing is always synchronous in single-user mode (MODE=SINGLE).

Value	Meaning
0	No asynchronous buffer flush (the default); flushing is synchronous.
1	The flush pool is to be sized at 25% of the buffer pool (LBP=) size.
n	The flush pool size is determined by the user.

When LFIOP=1 or LFIOP=n and the specified or calculated flush pool size is less than 80000 bytes, Adabas uses a size of 80000 bytes.

In environments running in 31-bit addressing mode, the I/O pool is allocated above the 16-megabyte line.

Example:

Enable asynchronous buffer flush operation for this session and allocate an I/O pool of 200,000 bytes (25% of the LBP size).

```
ADARUN PROG=ADANUC,LFIOP=1,LBP=800000
```

LFP : Length of Internal Format Buffer Pool

Parameter	Specify . . .	Minimum	Maximum	Default
LFP	the size of the Adabas internal format buffer pool.	6000	2147483647	12000

This pool stores internal format buffers used with Adabas read and update commands as described in the Adabas Command Reference documentation.

Each internal format buffer pool entry requires approximately $60 + (20 * n)$ bytes, where "n" equals the number of fields referenced in the command.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

In environments running in 31-bit addressing mode, the LFP pool is allocated above the 16-megabyte line.

Example:

Allocate 8,000 bytes for the Adabas internal format buffer pool for this session.

```
ADARUN PROG=ADANUC,LFP=8000
```

LI : Length of Table of ISNs (TBI)

Parameter	Specify ...	Minimum	Maximum	Default
LI	the size of the TBI.	2000	16777215	10000

LI specifies the number of bytes allocated for the table of ISNs (TBI), which is used to store ISN lists (overflow ISNs or saved ISN lists). Each ISN list is identified by a command ID (CID) and stored in a TBI element.

You must also specify a value for the NSISN parameter, which sets the maximum number of ISNs per TBI element. Each TBI element has a length of $64 + (NSISN * 4)$ bytes.

The number of TBI elements required determines the length of the TBI. For complex searches, the Adabas nucleus builds temporary TBI elements in addition to the permanent TBI element; for example, for holding the results of descriptor searches, for reading the result records in Data Storage and collecting matches, or for holding the results of the partial criteria associated with the "R" operator. Although the temporary TBI elements are deleted at the end of a search, the TBI must be large enough to hold all of the temporary and permanent TBI elements in parallel during search processing.

The Adabas session statistics or Adabas Online System can be used to set this parameter for the next session.

When running with Adabas expanded files, a larger-than-normal LI value is recommended.

In environments running in 31-bit addressing mode, the LI pool is allocated above the 16-megabyte line.

Example:

Allocate 30,000 bytes for the table of ISN list IDs for this session.

```
ADARUN PROG=ADANUC,LI=30000
```

LOCAL : Local Nucleus

Parameter	Specify . . .	Possible Values	Default
<u>LOCAL</u>	whether a nucleus is isolated and available for local use only.	YES NO	YES

Defines an isolated nucleus that is only available locally. The nucleus is unknown to Entire Net-Work. A nucleus specifying LOCAL=YES (the default) can have the same database ID as another database nucleus on another network node.

Value	Meaning
YES	Isolates this nucleus (that is, makes it unaddressable) from other Entire Net-Work nodes.
NO	Allows the nucleus to receive calls from other Entire Net-Work nodes.

Example:

The Adabas nucleus is isolated and can not be addressed by other Entire Net-Work nodes.

```
ADARUN PROG=ADANUC,LOCAL=YES
```

LOGGING / LOGxx : Command Logging Control

Parameter	Specify . . .	Possible Values	Default
<u>LOGGING</u>	whether to log Adabas commands.	YES NO	NO

The LOGGING parameter specifies whether to use command logging for the Adabas session. The LOGxx parameters specify the type of information to be logged.

If you specify LOGGING=YES, you must also specify YES for one or more of the following ADARUN parameters. By default, each type of information is *not* logged.

Parameter	Specify whether to log the ...	Possible Values	Default
LOGCB	Adabas control block	YES NO	NO
LOGCLEX	command log extension (CLEX) *	YES NO	NO
LOGFB	format buffer	YES NO	NO
LOGIB	ISN buffer	YES NO	NO
LOGIO	I/O activity	YES NO	NO
LOGRB	record buffer	YES NO	NO
LOGSB	search buffer	YES NO	NO
LOGUX	user exit B data *	YES NO	NO
LOGVB	value buffer	YES NO	NO

* LOGUX=YES and LOGCLEX=YES can only be used if CLOGLAYOUT=5 (the default) is also used. LOGCLEX=YES also requires REVIEW=NO although the Adabas Review modules must be available. Command log records with CLEX can be viewed using the batch program REVIEWB.

You can use Adabas operator commands or Adabas Online System to modify logging parameters during session execution.

Example:

Run the Adabas nucleus with command logging; log the Adabas control block.

```
ADARUN PROG=ADANUC,LOGGING=YES,LOGCB=YES
```

LOGSIZE : Maximum Command Log Size

Parameter	Specify . . .	Minimum	Maximum	Default
LOGSIZE	the maximum blocksize for the sequential command log DD/LOG.	100	32 760	value of QBLKSIZE

This parameter controls the maximum block size for the sequential command log DD/LOG. The default value is identical to the value of the QBLKSIZE parameter. If dual or multiple command logging is used, this parameter is ignored.

Example:

The blocksize of the command log is 2,000 bytes.

```
ADARUN PROG=ADANUC,LOGSIZE=2000
```

LP : Length of Data Protection Area (Work Part 1)

Parameter	Specify . . .	Minimum	Maximum	Default
LP	the number of blocks to allocate to Work part 1.	200	see text	1000

The LP parameter specifies the number of blocks to allocate to the Work data protection area, which is part 1 of the Work dataset. This area must be large enough to accommodate the data protection information for the current transaction for all ET logic users. Details can be found in *Adabas Restart and Recovery*.

The LP parameter is no longer limited to 65535 blocks; however, it is still limited by the size of the Work dataset and the sizes of Work parts 2, 3, and 4.

Note:

For any pending autorestart processing, Adabas retains and uses the value of the LP parameter from the previous session. If the LP parameter for the new session is specified with a different value than in the previous session, the nucleus will process any autorestart using the old value of LP, and establish the new LP size at the end of the autorestart.

Example:

Use 2,000 blocks to store data protection information on the Adabas Work dataset.

```
ADARUN PROG=ADANUC,LP=2000
```

LQ : Length of Sequential Command Table

Parameter	Specify . . .	Minimum	Maximum	Default
LQ	the size of the table of sequential commands.	2000	16 777 215	10 000

The table of sequential commands contains internal entries required during the processing of the Adabas read sequential (L2/L5, L3/L6, and L9) commands.

Each sequential pass of an Adabas file requires one entry in this table. Each table entry requires approximately 120 bytes. An entry is deleted when an end-of-file condition is detected during sequential processing or when an RC or CL command is issued.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session. If using expanded files, the LQ value may need to be set to a higher-than-normal value.

The LQ value also determines the maximum number of command IDs that each user can specify. See the NQCID parameter description, later in this chapter, for more information.

In environments running in 31-bit addressing mode, the LQ pool is allocated above the 16-megabyte line.

Example:

Allocate 30,000 bytes for the table of sequential commands for this session.

```
ADARUN PROG=ADANUC,LQ=30000
```

LRDP : Length of the Cluster Block Update Redo Pool

Parameter	Specify . . .	Minimum	Maximum	Default
<u>LRDP</u>	the size (in bytes) of the cluster block update "redo" pool.	0; or 80,000	2,147,483,647	LFIOP value

With deferred caching, the publication of updated blocks to the cache may fail due to conflicting updates to the same blocks by other nuclei in the cluster. Every cluster nucleus is therefore capable of redoing the updates it has not yet written to the cache. The nucleus maintains information about these updates in the "redo" pool.

Otherwise, with deferred caching enabled, the minimum pool size is 80,000 bytes and the (theoretical) maximum size is 2,147,483,647 bytes. If no value is specified, the nucleus takes as default for LRDP the value of the LFIOP parameter).

Different nuclei in the same cluster can have different settings of LRDP. It is also possible, although not recommended, to run one nucleus with LRDP=0 and a peer nucleus with LRDP>0.

If the redo pool specification is too small and the pool runs full, the nucleus is forced to write updated blocks to the cache before the end of transaction. Such additional cache writes will impair system performance.

Parameter Dependencies

LRDP uses the value of the LFIOP parameter as its default. The LFIOP parameter enables asynchronous buffer flush operation and sets the I/O pool size.

The LRDP parameter is only effective in an Adabas cluster nucleus; that is, CLUSTER={SYSPLEX | LOCAL} and NUCID is nonzero.

Pool Usage Reports

The nucleus reports on the usage (high watermark) of the redo pool in a shutdown statistic and in the response to the DRES command from the operator console or from ADADBS OPERCOM.

LS : Length of Internal Sort Area

Parameter	Specify . . .	Minimum	Maximum	Default
LS	the maximum number of bytes to be used for internal sort processing.	19 968	see note	49 920

Note:

To ensure that a single search command does not use too much of the work pool space, the maximum LS value must be less than or equal to the LWP (work pool length) minus the minimum LS value (19968) divided by two ($LS \leq (LWP - 19968) / 2$). See the *LWP parameter*.

The LS parameter specifies the maximum number of bytes within the Work pool to be used for sort work space during the processing of a single command of the following type:

- S2 and S9 commands;
- complex search commands, which require the storage of bit strings.

The following formula may be used to estimate the number of Adabas ISNs that may be sorted in the work space provided (S2 or S9 commands):

Notes:

1. TOPISN is the highest ISN currently in use for the file, less (MINISN - 1). TOPISN is not controlled by a parameter.
2. 4-byte ISNs are used for the internal sort area, regardless of the ISNSIZE specified for the file.

If the TOPISN in the file is less than 65536:

using 1 descriptor LS/12
 using 2 descriptors LS/16
 using 3 descriptors LS/20

If the TOPISN in the file is greater than 65536:

using 1 descriptor LS/14
 using 2 descriptors LS/20
 using 3 descriptors LS/26

For complex searches, it is desirable for performance reasons to set LS to the next integer multiple of 256 bytes greater than $TOPISN / 8$.

For example, if $TOPISN = 480000$, then $TOPISN / 8 = 60000$, so LS should be set to the next higher multiple of 256, namely 60160.

Example:

The maximum amount of internal sort space during this session is 60,160 bytes.

```
ADARUN  PROG=ADANUC,LS=60160
```

LU : Length of Intermediate User Buffer Area

Parameter	Specify . . .	Minimum	Maximum	Default
LU	the size of the intermediate user buffer area.	4000 (see note 1)	65 535 (see note 2)	65 535 (see note 3)

Notes:

1. The minimum value may be less if a lesser value was specified by the DBA using Adabas Online System.
2. The maximum value must be smaller than the byte count implied by the NAB (number of attached buffers) parameter. For example, the NAB default of 16 allocates more than 64KB. See ADARUN parameter NAB.
3. Due to the length of the record buffer of the utilities that need the nucleus, e.g., ADAULD, the default value is set to 65,535. If the value of LU is less than 65,535 for an Adabas session, a response code will occur when such a utility is running.

The size specified must be large enough to accommodate all Adabas control information (108 bytes), all user buffers (format, record, search, value, ISN) that may be required for any given Adabas command, plus any user information passed from Adabas link routines to nucleus user exits.

If the multifetch/prefetch option or a utility that needs large record/ISN buffers is to be used during the session, the setting of LU must be large enough to contain the buffers needed.

Example:

Run the Adabas nucleus with an Adabas intermediate user buffer area of 20,000 bytes.

```
ADARUN PROG=ADANUC,LU=20000
```

LWKP2 : Length of ISN List Processing Area (Work Part 2)

Parameter	Specify . . .	Minimum	Maximum	Default
<u>LWKP2</u>	the number of Work blocks to use for intermediate ISN lists.	0	16 777 215	0

This parameter specifies the size (in blocks) of the Adabas Work part 2, which holds intermediate ISN lists during Sx commands.

If LWKP2=0 (the default), Adabas calculates the Work part 2 size automatically using the following formula:

$$\text{WORK2} = 22 + 2 * (4 * \text{RECORDS} / (\text{BLKSIZE} - 16))$$

where

WORK2 is the Work part 2 space requirement in blocks.

RECORDS is the number of records in the file with the most records. This number equals $\text{TOPISN} - \text{MINISN} + 1$, where **TOPISN** is the highest ISN currently used in the file. **MINISN** is the lowest ISN used in the file. The **MINISN** value is specified with the ADACMP/ADALOD parameter **MINISN**; 1 is the default. You can use the ADAREP utility to display the **TOPISN** and **MINISN** values for the files in a database.

BLKSIZE is the block size of the device where the Work dataset is stored (see the Adabas Installation documentation).

If you choose not to use LWKP2=0, the minimum value is LWKP2=20 blocks.

Note:

An Adabas internal table requires one byte of storage for each Work part 2 block.

Example:

For this Adabas session, the size of the Adabas Work part 2 is 50,000 blocks.

```
ADARUN PROG=ADANUC,LWKP2=50000
```

LWP : Length of Adabas Work Pool

Parameter	Specify . . .	Minimum	Maximum	Default
LWP	the size of the Adabas work pool.	80 000 but not less than NT * 25000	2147483647	150 000

Note:

The LWP (work pool length) parameter value must be equal to or greater than the LS (internal sort area length) value plus its minimum value (19968) multiplied by 2; that is, $((LS + 19968) * 2)$.

The Adabas work pool area is used to store the following:

- descriptor value tables (DVT);
- intermediate user buffer areas (see parameter LU);
- sort work space areas (see parameter LS);
- work space during session and command execution.

In environments running in 31-bit addressing mode, the work pool space is allocated above the 16-megabyte line.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session. If expanded files are being used, the LWP value may need to be increased above the normal setting.

Example:

Run the Adabas nucleus with a work pool of 900,000 bytes.

```
ADARUN PROG=ADANUC,LWP=900000
```

MODE : Mode of Operation

Parameter	Specify . . .	Possible Values	Default
<u>MODE</u>	the Adabas operating mode.	SINGLE MULTI	MULTI

Value Meaning

SINGLE	Run in single-user mode: Load the Adabas nucleus and the user program or Adabas utility into the same region.
MULTI	Run in multiuser mode: Load only the ADALNK routine into the same region as the program/utility. Load the Adabas nucleus into a different region. ADALNK will manage all interregion communication between the program or utility and the nucleus.

Note the following:

1. The MODE parameter does not affect parallel utility execution.
2. The nucleus cannot run in single-user mode (MODE=SINGLE cannot be specified) if it is to run with
 - Adabas Parallel Services or Adabas Cluster Services (NUCID 1)
 - support for distributed transactions (DTP=RM)
 - as an Adabas Transaction Manager (DTP=TM)
3. When running several nuclei against the same database, only one update nucleus can be used.

Example:

Execute the ADAINV utility in single-user mode.

```
ADARUN  PROG=ADAINV,MODE=SINGLE
```

MSGBUF : Size of Message Buffer

Parameter	Specify . . .	Minimum	Maximum	Default
<u>MSGBUF</u>	the size of the message buffer area.	0	128	0

MSGBUF specifies the size (in kilobytes) of the area used to buffer messages for use for performance and tuning and problem analysis. Adabas messages buffered in this area can be reviewed even when online access to the console or DDPRINT messages is unavailable. While the buffer is intended for messages written to the system log or DDPRINT, it can be used for any purpose by any caller.

MSGBUF presupposes that ADARUN SMGT=YES.

MSGBUF=0 (the default) means that message buffering is not active for the session.

To activate message buffering, set MSGBUF to a value greater than 0. Software AG recommends setting MSGBUF to 10 (kilobytes).

Message buffering can be temporarily disabled and subsequently reactivated using the SMGT operator command.

See the Adabas DBA Tasks documentation for more information about message buffering.

Example:

Activate message buffering when the Adabas nucleus is initialized with a message buffer size of 10 KB.

```
ADARUN PROGRAM=ADANUC
ADARUN SMGT=YES
ADARUN MSGBUF=10
```

MSGCONSL : Case of Messages Sent to the Console

MSGDRUCK : Case of Messages Sent to DD/DRUCK

MSGPRINT : Case of Messages Sent to DD/PRINT

Parameter	Specify the case of Adabas . . .	Possible Values	Default
<u>MSGCONSL</u>	messages sent to the console	<u>UPPER</u> <u>MIXED</u>	<u>MIXED</u>
<u>MSGDRUCK</u>	messages sent to the DD/DRUCK dataset		
<u>MSGPRINT</u>	messages sent to the DD/PRINT dataset		

Adabas messages are displayed in mixed upper-/lowercase. You can use one or more of these ADARUN parameters to display different message types in all uppercase.

Note:

The first message line in DDPRINT will not be influenced by the value of parameter MSGPRINT.

Example:

Display messages in the DDDRUCK and DDPRINT datasets in all uppercase:

```
ADARUN MSGDRUCK=U,MSGPRINT=U
```

MXMSG-Timeout Threshold for Internucleus Communication

Parameter	Specify . . .	Minimum	Maximum	Default
MXMSG	the timeout threshold for messages between cluster nuclei (in seconds)	1	32767	60

The MXMSG parameter is optionally used to set the maximum time allowed in seconds for commands from one cluster nucleus to another in a parallel sysplex environment using XCF.

In case of an XCF communication failure, it may take as much time as set in this parameter until the failure is noticed by the cluster nuclei.

If an internucleus command is not acknowledged within this timeframe, the sending nucleus generates a response code 125 and, in most cases, terminates abnormally (ABEND).

MXTNA : Maximum Non-activity Time

Parameter	Specify . . .	Minimum	Maximum	Default
MXTNA	the maximum non-activity time limit that can be specified in the OP command for an individual user.	1	65535	3600

The TNAX parameter specifies the maximum elapsed time (in units of 1.048576 seconds) that users may be considered active without issuing an Adabas command. Limits may be set for three types of user:

- the TNAA parameter for access-only users
- the TNAE parameter for ET logic users
- the TNAX parameter for exclusive control users

You can override these parameter settings for an individual user by specifying a different value in the Adabas control block of an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.

For these parameters, the MXTNA parameter sets a limit on the amount of time that can be specified in the OP command for an individual user.

If the nonactivity time limit is exceeded, action is taken as described in *Adabas Timeout Feature*.

The value of the MXTNA parameter can only be set when the nucleus is started. It cannot be changed using operator commands or the ADADBS OPERCOM utility function.

Example:

During this Adabas session, the nonactivity time limit for exclusive control users is about 5 minutes (300 seconds). This limit cannot be extended for an individual user beyond 8 minutes (480 seconds).

```
ADARUN PROG=ADANUC ,TNAX=300 ,MXTNA=480
```

MXTSX : Maximum Sx Command Execution Time

Parameter	Specify . . .	Minimum	Maximum	Default
<u>MXTSX</u>	the maximum Sx command execution time limit that can be specified in the OP command for an individual user.	1	65535	3600

The TLSCMD parameter specifies the maximum time (in units of 1.048576 seconds) to be used to process a single Adabas S1, S2, or S4 command with a complex search criterion.

If the estimated time to process a command exceeds the TLSCMD time limit, the command is not processed and the user who issued the command receives a nonzero response code.

If the estimated time to process a command does not exceed the time limit, the command is processed to completion, whether the actual time required exceeds the time limit specified by the TLSCMD or not.

You can override the TLSCMD parameter setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.

The MXTSX parameter sets a limit on the amount of time that can be specified in the OP command for an individual user.

The value of the MXTSX parameter can only be set when the nucleus is started. It cannot be changed using operator commands or the ADADBS OPERCOM utility function.

Example:

During this Adabas session, the Sx command execution time limit is about 1 hour 20 minutes (4800 seconds). This limit cannot be extended for an individual user beyond 2 hours (7200 seconds).

```
ADARUN PROG=ADANUC, TLSCMD=4800, MXTSX=7200
```

MXTT : Maximum Transaction Time

Parameter	Specify . . .	Minimum	Maximum	Default
MXTT	the maximum transaction time limit that can be specified in the OP command for an individual user.	1	65535	3600

The TT parameter specifies the maximum elapsed time (in units of 1.048576 seconds) permitted for a logical transaction issued by an ET logic user. See ADARUN parameter TT for more information.

You can override this parameter setting for an individual user by specifying a different value in the Adabas control block of an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.

The MXTT parameter sets a limit on the amount of time that can be specified in the OP command for an individual user.

The time measurement for a logical transaction starts when the first command that results in a record being placed in hold status is issued. It ends when an ET, BT, or CL command is issued.

If the transaction time limit is exceeded, action is taken as described in *Adabas Timeout Feature* .

The value specified for MXTT directly influences the required size of the Adabas parameter LP (length of the data protection area).

Software AG recommends that you set the MXTT parameter lower than the MXTNA parameter.

The value of the MXTT parameter can only be set when the nucleus is started. It cannot be changed using operator commands or the ADADBS OPERCOM utility function.

Example:

During an Adabas session, the transaction time limit for ET logic users is about 3 minutes (180 seconds) and cannot be extended for an individual user beyond 5 minutes (300 seconds).

```
ADARUN  PROG=ADANUC,TT=180,MXTT=300
```

NAB : Number of Attached Buffers

Parameter	Specify ...	Minimum	Maximum	Default
<u>NAB</u>	the number of attached buffers to be used.	0	500,000	16

An attached buffer is an internal buffer used for interregion communication. It is required in all environments. Adabas allocates an attached buffer pool with a size equal to the value of NAB multiplied by 4096 bytes.

You may specify as many attached buffers as fit into the available virtual storage.

Note:

If a working NAB value is decreased for some reason, the related LU (length of intermediate user buffer) parameter value may also need to be decreased. The LU value cannot be greater than the byte count implied by the NAB value. Such LU / NAB changes may be needed or desired when reallocating a smaller NAB area.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

In environments running in 31-bit addressing mode, the attached buffer pool space is allocated above the 16-MB line.

Example:

Run the Adabas nucleus with 50 attached buffers.

```
ADARUN PROG=ADANUC ,NAB=50
```

NC : Number of Command Queue Elements

Parameter	Specify . . .	Minimum	Maximum	Default
NC	the maximum number of command queue elements.	20	32 767	200

The number of command queue elements (CQEs) established for the Adabas session determines the maximum number of Adabas commands that may be queued and/or be in process at any one time during the Adabas session.

Each Adabas command is assigned a CQE. The CQE is released when the user has received the results of the command or the user has been timed out.

192 bytes are required for each CQE.

Software AG recommends that you set NC high enough to allow one command per active user for possible synchronization during execution of the online SAVE database function of the ADASAV utility.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

Example:

Run the Adabas nucleus with a maximum of 500 elements in the command queue.

```
ADARUN PROG=ADANUC,NC=500
```

NCLOG : Number of Command Log Datasets

Parameter	Specify . . .	Minimum	Maximum	Default
NCLOG	the maximum number of command log datasets to be opened and logged in the PPT.	0, 2	8	0

This parameter specifies the maximum number of command log datasets to be opened and logged in the parallel participant table (PPT) for the current nucleus. Valid values are 0 (the default) and 2-8. Other values produce a parameter error. A zero value disables the parameter.

If more than the number of log datasets specified in the NCLOG parameter are allocated in the startup JCL, only the number specified in the NCLOG parameter are opened and logged; any additional allocated datasets are activated using the dynamic allocation process.

NCLOG must be specified with a nonzero value in conjunction with the CLOGDEV, CLOGSIZE, and UEX12 parameters.

NCLOG is not valid if DUALCLD, DUALCLS, DUALPLD, DUALPLS, or UEX2 are specified.

UEX2 and UEX12 are mutually exclusive.

If a nonzero NCLOG, CLOGDEV, and CLOGSIZE are specified, but LOGGING=NO, Adabas attempts to open and log the number of datasets specified by NCLOG; however, no logging occurs until the LOGGING parameter has been set to YES using either an operator command or Adabas Online System.

Example:

Run the Adabas nucleus with command logging. Use 6 command log datasets. Use a model 3380 device for command logging. The size of each command log dataset is 1350 blocks, based on an allocation of 10 cylinders. (Each cylinder of a 3380 device has 15 tracks, and each track can contain 9 blocks of command log data. Thus, 10 cylinders = 10 15 9 = 1350 blocks).

```
ADARUN  PROG=ADANUC ,LOGGING=YES ,NCLOG=6 ,CLOGDEV=3380 ,CLOGSIZE=1350
```

NH : Number of Hold Queue Elements

Parameter	Specify . . .	Minimum	Maximum	Default
NH	the maximum number of hold queue elements.	20	16777215	500

One hold queue element (HQE) is required for each record (ISN) placed in hold status. An HQE is released when the record is released from hold status.

The hold queue is normally used to hold ISNs for the duration of a transaction. Therefore NH should be large enough to contain at least the average number of ISNs held per transaction multiplied by the maximum number of transactions running in parallel.

Each HQE requires 28 bytes.

In environments running in 31-bit addressing mode, the hold queue is allocated above the 16-megabyte line.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

Example:

Run the Adabas nucleus with a maximum of 2000 elements in the hold queue.

```
ADARUN PROG=ADANUC,NH=2000
```

NISNHQ : Number of ISNs in Hold Queue per User

Parameter	Specify . . .	Minimum	Maximum	Default
<u>NISNHQ</u>	the maximum number of records that can be placed in hold status at the same time by one user.	1	(NH / 4) or 65535, whichever is smaller	(NH / NU) * 2 or 20, whichever is larger

NISNHQ determines the maximum number of held ISNs the hold queue (see the NH parameter description) can contain from one user. The most ISNs allowed on hold for a single user is 1/4 of the hold queue size, but not more than 65,535 ISNs. The default is the greater of either 20 or the number of ISNs the hold queue can contain divided by the number of users, multiplied by two.

If a user attempts to place more records in hold status than permitted, the user receives a nonzero response code even though there may still be space in the hold queue.

Example 1:

The maximum number of records that may be in hold status for a single user is 100.

```
ADARUN PROG=ADANUC,NISNHQ=100,NH=450...
```

Example 2:

NISNHQ is not specified. The maximum number of records that may be in hold status for a single user is 40 (the NISNHQ default of (NH / NU) * 2, which is greater than the alternative default of 20).

```
ADARUN PROG=ADANUC,NU=50,NH=2000...
```

Overriding the Parameter Setting

You can override this parameter setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.

NONDES : Nondesoriptor Searches

Parameter	Specify . . .	Possible Values	Default
NONDES	whether to allow (YES) or disallow (NO) searches using nondescriptors in search buffers.	YES NO	YES

By default, you can perform searches on Adabas fields that have not been defined as descriptors. However, nondesoriptor searches carry significant overhead.

NONDES=NO prevents searches on nondescriptors. An attempt to search on a nondesoriptor field produces response code 68.

Example:

Disallow searches using nondescriptors.

```
ADARUN PROG=ADANUC, NONDES=NO
```

NPLOG : Number of Protection Log Datasets

Parameter	Specify . . .	Minimum	Maximum	Default
NPLOG	the maximum number of protection log datasets to be opened and logged in the PPT.	0, 2	8	0

This parameter specifies the maximum number of protection log datasets to be opened and logged in the parallel participant table (PPT) for the current nucleus. Valid values are 0 (the default) and 2-8. Other values produce a parameter error. A zero value disables the parameter.

If more than the number of log datasets specified in the NPLOG parameter are allocated in the startup JCL, only the number specified in the NPLOG parameter are opened and logged; any additional allocated datasets are activated using the dynamic allocation process.

NPLOG must be specified with a nonzero value in conjunction with the PLOGDEV, PLOGSIZE, and UEX12 parameters.

NPLOG is not valid if DUALCLD, DUALCLS, DUALPLD, DUALPLS, or UEX2 are specified.

UEX2 and UEX12 are mutually exclusive.

Example:

Run the Adabas nucleus with 6 data protection log datasets. Use a model 3380 device; the size of each log dataset is 1200 blocks, based on an allocation of 10 cylinders. (Each cylinder of a 3380 device has 15 tracks, and each track contains 8 blocks of protection log data. Thus, 10 cylinders = $10 * 15 * 8 = 1200$ blocks).

```
ADARUN PROG=ADANUC ,NPLOG=6 ,PLOGDEV=3380 ,PLOGSIZE=1200
```

NQCID : Number of Active Command IDs per User

Parameter	Specify . . .	Minimum	Maximum	Default
NQCID	the maximum number of active command IDs per user.	1	65535	20

The table of sequential commands contains the control blocks for sequential read operations. Each operation is identified by a command ID (CID). NQCID defines the total number of CIDs allowed per user, and therefore the maximum number of table elements the user has available at any point in time.

Note:

The NQCID value specified does not ensure that all users have the specified count of CIDs available at all times. The actual number of CIDs available to a single user depends on the totals of table elements available and sequential operations active at a given time.

Example:

The maximum number of parallel sequential processes per user is 30.

```
ADARUN PROG=ADANUC ,NQCID=30 ,NU=200 ,LQ=3000
```

Since one element in the sequential command table occupies approximately 60 bytes (see the description of the LQ parameter), NQCID=30 limits each user to 1800 bytes of the table.

Overriding the Parameter Setting

You can override this parameter setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.

NSISN : Number of ISNs per ISN Table Element

Parameter	Specify . . .	Minimum	Maximum	Default
NSISN	the maximum number of ISNs per element in the table of ISNs (TBI).	7*	see text	51

* The minimum value may be lower if specified using Adabas Online System.

The parameter NSISN specifies the maximum number of ISNs kept in a TBI element in memory (list of resulting ISN lists). The more ISNs that are kept in memory, the fewer the Work I/Os needed to read additional resulting ISN lists from the Work dataset. This parameter influences the performance of the L1/4 command with the "N" (GET NEXT) option.

The highest value that can be specified for NSISN depends on the Work device. For Adabas-specified Work block sizes, the maximum values are shown in the following tables:

The following tables do not apply to user-defined Work block sizes; in this case, the maximum is determined as follows, where WORK is the block size of the Work dataset:

- $(\text{WORK} - 6) / 4$
- The TBI element has a length of $64 + (\text{NSISN} * 4)$ bytes in the TBI pool (see the LI parameter).

Although supported, the BS2000 device type 2007 is not recommended for use with Adabas. Support for the 2007 will be removed in a later Adabas release.

IBM and Compatible Devices

Device Type	Max. Value	Device Type	Max. Value
3310	1022	8345	5728
3330	1061	8350	2358
3340	877	8380	2267
3350	1155	8381	2867
3359	1155	8385	5865
3370	1278	8390	2674
3375	1022	8391	3418
3379	1022	8392	4611
3380	1371	8393	6995
3389	1371	9332	1278
3390	1429	9335	1278
		9345	2785

BS2000 Devices

Device Type	Max. Value	Device Type	Max. Value
2000	1022	2010	4096
2001	1022	2200	4096
2002	2046	2201	3072
2003	1534	2202	4096
2004	2558		
2005	2046		
2006	2558		
2007	7678		
2008	8188		
2009	8188		

Example:

The maximum number of ISNs per TBI element during the Adabas session is 400.

```
ADARUN PROG=ADANUC,NSISN=400
```

Overriding the Parameter Setting

You can override this parameter setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.

NT : Number of Threads

Parameter	Specify . . .	Minimum	Maximum	Default
NT	the number of user threads to be used during the Adabas session.	3	250	5

A thread is used to execute a single command for any user (access or update).

Each thread requires 8192 bytes.

In environments running in 31-bit addressing mode, the threads are allocated above the 16-megabyte line.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

Example:

Run the Adabas nucleus with 6 user threads.

```
ADARUN PROG=ADANUC,NT=6
```

NU : Number of User Queue Elements

Parameter	Specify . . .	Minimum	Maximum	Default
NU	the maximum number of user queue elements.	20	16 777 215	200

Adabas assigns a user queue element (UQE) to each active user when the user issues an OP command or (if the user did not issue an OP command) the first Adabas command.

Adabas releases the UQE when the user issues a CL command or when there is a timeout for an ET logic user who did not specify a restart ID in the control block (additions 1 field) for the OP command.

Currently, $272 * (NU + 4)$ bytes are allocated to the user queue. The length of each UQE is 288 bytes.

In environments running in 31-bit addressing mode, the user queue is allocated above the 16-megabyte line.

Because each user's file list can be a different size, the number of users who can be active at the same time may be less than the NU value. Therefore, you may need to specify an NU value that is greater than the maximum number of users desired.

The Adabas session statistics or Adabas Online System can be used to tune this parameter for the next session.

Example:

Run the Adabas nucleus with a maximum of 500 elements in the user queue.

```
ADARUN PROG=ADANUC,NU=500
```

NUCID : Adabas Cluster Nucleus ID

Note:

The NUCID parameter replaces the earlier parameters PLXID and SMPID.

Parameter	Specify . . .	Possible Values	Default
NUCID	the ID for a nucleus in an Adabas cluster.	0-65000	0

NUCID identifies an Adabas cluster nucleus. The cluster may span operating system images under Adabas Cluster Services, or it may be confined to a single operating system image under Adabas Parallel Services. A value greater than 0 (the default) identifies this nucleus as an Adabas cluster nucleus.

Value	Meaning
0	The default setting. This is not an Adabas cluster nucleus. For ADADBS OPERCOM and AOS, the NUCID=0 command option means "global", indicating that the command applies to all nuclei in the cluster.
1-65000	This is an Adabas cluster nucleus. A nonzero value automatically sets the parameter READONLY=NO; MODE=MULTI should also be specified. Values need not be specified contiguously. This value must be unique among all Adabas database IDs, NUCIDs, Natural buffer pool IDs, etc. Entire Net-Work target IDs (suggested range 65482-65535), and internal target IDs used by Adabas Cluster Services (restricted range 65001-65481) are assigned outside of the range of possible values for NUCID. For ADADBS OPERCOM and AOS, a nonzero value for the NUCID command option indicates that the command applies just to the cluster nucleus specified.

Note:

For ADADBS OPERCOM and AOS, a zero value for the NUCID command option indicates that the command applies to all nuclei in the cluster ("global"). A nonzero value for the NUCID command option indicates that the command applies only to the cluster nucleus specified.

Guidelines

- Each nucleus being run must have its own Work dataset.
- All nuclei in a cluster must be run against the same database.
- Software AG strongly recommends that you run utilities or user applications against a nucleus in the local OS/390 image. Although a remote open nucleus may be used, performance degradation due to network overhead is likely.
- If ADACOM is active for a database and you start a nucleus with NUCID=0 for that database, any utilities or user applications sending commands to that nucleus receive a response code 148.

Example:

Start an Adabas cluster session with the specified cluster nuclei:

```
ADARUN PROG=ADANUC,NUCID=1, ...
```

OPENRQ : Open Command Required

Parameter	Specify ...	Possible Values	Default
OPENRQ	whether an open command is required.	YES NO	YES

This parameter may be used to indicate whether an open (OP) command must be issued as the first command of a user session.

Value	Meaning
YES	The default setting. Any user attempting to issue commands to Adabas without previously issuing an OP command receives a response code.
NO	Users are not required to issue an OP as the first command of the session.

Example:

The first session command issued by users does not have to be an OP command.

```
ADARUN PROG=ADANUC,OPENRQ=NO
```

PLOGDEV : Multiple Protection Log Device Type

PLOGSIZE : Multiple Protection Log Size

Parameter	Specify . . .	Possible Values	Default
PLOGDEV	the device type to be used for multiple protection logs.	see text	value of DEVICE parameter
PLOGSIZE	the number of blocks available for each protection log dataset.	Min 16, Max 16 777 215	none

The PLOGDEV and PLOGSIZE parameters specify the type of external storage device used for multiple data protection log datasets and the number of blocks available for each log dataset. These parameters must be used *together*; if you specify one parameter, you must also specify the other parameter.

You must also specify the NPLOG parameter to use multiple protection logs. If you specify a user exit, it must be UEX12. See *Adabas Restart and Recovery* for information about multiple dataset protection logging.

The protection log device type must be the same as that of the Adabas Work dataset. The Adabas Installation documentation specifies the device types that Adabas supports.

All protection log datasets must be formatted using the ADAFRM utility before they are used. The appropriate JCL for the datasets is required.

Example:

Run the Adabas nucleus with 6 data protection log datasets. Use a model 3380 device; the size of each log dataset is 1200 blocks, based on an allocation of 10 cylinders. (Each cylinder of a 3380 device has 15 tracks, and each track contains 8 blocks of protection log data. Thus, 10 cylinders = $10 * 15 * 8 = 1200$ blocks).

```
ADARUN  PROG=ADANUC ,NPLOG=6 ,PLOGDEV=3380 ,PLOGSIZE=1200
```

PLOGRQ : Protection Log Required

Parameter	Specify . . .	Possible Values	Default
PLOGRQ	whether a protection log is required for this session.	YES NO SEL FORCE	YES

The valid values are as follows:

Value	Meaning
YES	The default setting. Any attempt to start an Adabas nucleus without a protection log causes the Adabas initialization to terminate with an error message.
NO	In this Adabas session: <ul style="list-style-type: none"> - multiple PLOG datasets are used if a PLOGSIZE parameter is provided; - dual PLOG datasets are used if a DUALPLS parameter is provided; - a SIBA dataset is used if JCL for a SIBA dataset is provided; - otherwise, no protection log dataset is used.
SEL	Selective protection logging is switched on only when running an ADASAV SAVE (database) or SAVE FILE= ... utility operation with an active nucleus. Protection logging is switched off when the ADASAV SAVE operation completes successfully.
FORCE	Any attempt to start an Adabas nucleus without UEX2 or UEX12, or without a protection log causes the Adabas initialization to terminate with an error message. Additionally, an error occurs if the initialization process determines that different PLOG datasets have been specified for the session and the old PLOG datasets have not yet been copied.

Example:

A data protection log (SIBA or PLOG) is not mandatory during the Adabas session but will be used if provided.

```
ADARUN PROG=ADANUC,PLOGRQ=NO
```

PREFETCH : Activate Prefetch Feature

Parameter	Specify . . .	Possible Values	Default
PREFETCH	whether to enable a prefetch option automatically.	YES OLD NO	NO

This parameter determines whether the Adabas prefetch feature is enabled automatically; if it is, execution is transparent to the user program.

There are two Adabas prefetch options: the multifetch option is supported across Adabas platforms; the prefetch option is available only on mainframe Adabas. For information about these two options, see the Adabas Command Reference documentation.

Records can be prefetched and stored in a prefetch buffer for the following commands:

- L1/L4 with I or N option (read by ISN, find with GETNEXT)
- L2/L5 (read physical)
- L3/L6 (read logical by descriptor)
- L9 (histogram)

Sequential commands of the types listed above can retrieve the next record from the prefetch buffer, which saves cross-region/cross-network calls to the nucleus.

The PREFETCH parameter has the following values:

Value	Meaning
YES	Adabas enables the command-level multifetch (M) option.
OLD	Adabas enables the command-level prefetch (P) option.
NO	Adabas does not enable the prefetch feature.

Note:

Specify PREFETCH=YES or PREFETCH=OLD only when MODE=MULTI (multiuser mode). Do not specify these settings when running Adabas utilities. Dynamic reversal of read direction for L3/L6/L9 is not supported with these settings.

Example 1:

Execute a user program using multifetch.

```
ADARUN PROG=USER,PREFETCH=YES
```

Example 2:

Execute a user program using prefetch.

```
ADARUN PROG=USER,PREFETCH=OLD
```

PREFICMD : Include Commands in Prefetch/Multifetch Processing

PREFIFIL : Include Files in Prefetch/Multifetch Processing

Parameter	Specify . . .	Possible Values	Default
PREFICMD	the commands to include in prefetch processing.	L1-L9	none
PREFIFIL	the files to include in prefetch processing.	1-5000	none

Together, PREFIFIL and PREFICMD specify one or more files to be included in prefetch/multifetch processing. PREFIFIL specifies one or more files by file number; PREFICMD specifies a command for each file number.

You can either "include" list of files using PREFIFIL/PREFICMD or "exclude" lists of files using the parameters PREFXFIL/PREFXCMD), depending on which list is shorter. You cannot both include and exclude files at the same time.

If prefetching/multifetching has been specified and

- files/commands have been neither included nor excluded, the prefetching/multifetching applies to all files by default.
- files/commands have been included using the ADARUN parameters PREFIFIL/PREFICMD, the prefetching/multifetching applies only to the files on the "inclusion" list.
- both PREFIFIL/PREFICMD and PREFXFIL/PREFXCMD are specified, Adabas recognizes the exclusion parameters PREFXFIL/PREFXCMD and ignores the inclusion parameters PREFIFIL/PREFICMD.

Example:

Include the L3 command for file 8 in prefetch processing.

```
ADARUN PREFIFIL=8 ,PREFICMD=L3
```

Multiple Inclusions

Specify multiple file numbers in multiple PREFIFIL statements or in a list, separated by commas. Specify multiple commands in multiple PREFICMD statements or in a list, separated by slashes (/). The maximum number of entries for each is 50.

If you use lists, they must have the same number of entries. There is a one-to-one correspondence, by position, between a file number in the PREFIFIL list and a command in the PREFICMD list.

Example:

Include the L3 command for file 8 and the L3 and L9 commands for file 12 in prefetch/multifetch processing.

```
ADARUN PREFIFIL=8,12,12,PREFICMD=L3/L3/L9
```

or

```
ADARUN PREFIFIL=8,PREFICMD=L3
    ADARUN PREFIFIL=12,PREFICMD=L3
    ADARUN PREFIFIL=12,PREFICMD=L9
```

As the preceding example shows, you can specify the same file number several times, with a different command each time. Likewise, you can specify the same command code several times, with a different file number each time.

Eligible Commands and Command Pairs

The following commands can be included in prefetch/multifetch processing for a file:

- L1/L4
- L2/L5
- L3/L6
- L9

If either command in one of the pairs above is specified in a PREFICMD statement, *both* commands are included. For example, if PREFICMD=L1 is specified, both L1 and L4 are included.

Example:

Execute a user program with multifetch and allocate 20,000 bytes for the total prefetch buffer area. Include the L3 and L6 commands for files 8 and 12 in multifetch processing. Include *all* commands for file 7.

```
ADARUN PROG=USER,PREFETCH=YES,PREFTBL=20000
    ADARUN PREFIFIL=8,12,PREFICMD=L3/L3
    ADARUN PREFIFIL=7,7,7,7,PREFICMD=L1/L2/L3/L9
```

No PREFSBL value is specified, so the buffer length for a single prefetch sequence defaults to 3,000 bytes; up to 6 prefetch sequences can be handled.

PREFNREC : Multifetch Record Count

Parameter	Specify . . .	Minimum	Maximum	Default
PREFNREC	the maximum count of records to be prefetched.	0	see text	see text

This parameter specifies the maximum count of records that the multifetch feature (specified by the PREFETCH=YES parameter) obtains.

The maximum that can be specified depends on the maximum that can be held by the record and ISN buffers, each of which is assumed to be less than 32 kilobytes.

If no value or zero (the default) is specified, the sizes of the ISN and record buffers determine the number of records that are prefetched. Specifying zero (0) *does not* disable multifetch operation.

Example:

Enable the multifetch feature and allow a maximum of 32 records to be prefetched.

```
ADARUN PROG=USER , PREFETCH=YES , PREFNREC=32
```

PREFSBL : Prefetch Buffer Length per Call

Parameter	Specify . . .	Minimum	Maximum	Default
PREFSBL	the prefetch buffer length for one prefetch.	1024	65535	3000

The PREFSBL parameter specifies the size of the buffer for a single prefetch, which determines how much record data is prefetched. See also the description of the PREFTBL parameter, which specifies the total buffer space used by prefetch/multifetch.

When prefetch or multifetch is active, Adabas makes a buffer available to the user program for storing prefetched records. If a required record is already in a prefetch buffer, the record is taken directly from the buffer. PREFSBL specifies the size of this buffer.

Note:

Do not specify this operand unless PREFETCH=YES or PREFETCH=OLD also is specified.

PREFSBL must specify at least *twice the size of the largest record* that might be prefetched. Note that unnecessary prefetching can cause needless accesses. In addition, the nucleus keeps held ISNs for all unprocessed prefetched records in hold status following a BT or ET command.

For long sequential processes, this parameter should be as large as possible unless the sequential processes are frequently interrupted with "value start" (L3/L6) or RC commands; in that case, a large value often causes records to be prefetched needlessly.

Example:

Execute a user program using multifetch. Use a 15,000-byte prefetch buffer.

```
ADARUN PROG=USER,PREFETCH=YES,PREFSBL=15000
```

PREFTBL : Total Size of Prefetch Buffer

Parameter	Specify . . .	Minimum	Maximum	Default
<u>PREFTBL</u>	the size of the total prefetch buffer area.	6144	16777215	30000

PREFTBL specifies the size of the total prefetch buffer area for this operation. This buffer length should be large enough to hold the PREFSBL-specified value, multiplied by the number of command-ID-controlled sequential prefetches for this operation. This allows a user program to perform multiple sequential operations requiring prefetches, each with a prefetch buffer of the size specified in the PREFSBL parameter.

If this user program has a single prefetch sequence only, the PREFTBL value must be equal to or greater than the PREFSBL value. The PREFTBL-specified space is always allocated to the program.

Note:

Do not specify this operand unless PREFETCH=YES or PREFETCH=OLD is also specified.

Example:

Execute a user program using multifetch. No PREFSBL value is specified, so use the default buffer length for a single prefetch (3,000 bytes). Allocate 30,000 bytes for the total prefetch buffer area; this size provides buffer space for up to 10 prefetch sequences.

```
ADARUN PROG=USER , PREFETCH=YES , PREFTBL=30000
```

PREFXCMD (PREXFIL) : Exclude Commands (Files) from Prefetch/Multifetch Processing

Parameter	Specify ...	Possible Values	Default
PREFXCMD	the commands to exclude from prefetch processing.	L1-L9	none
PREXFIL	the files to exclude from prefetch processing.	1-5000	none

Together, PREXFIL and PREFXCMD specify one or more files to be excluded from prefetch/multifetch processing. PREXFIL specifies one or more files by file number; PREFXCMD specifies a command for each file number.

You can either "include" list of files using PREFIFIL/PREFICMD) or "exclude" lists of files using the parameters PREXFIL/PREFXCMD, depending on which list is shorter. You cannot both include and exclude files at the same time.

If prefetching/multifetching has been specified and

- files/commands have been neither included nor excluded, the prefetching/multifetching applies to all files by default.
- files/commands have been excluded using PREXFIL/PREFXCMD, the prefetching/ multifetching applies only to the files on the "exclusion" list.
- both PREFIFIL/PREFICMD and PREXFIL/PREFXCMD are specified, Adabas recognizes the exclusion parameters PREXFIL/PREFXCMD and ignores the inclusion parameters PREFIFIL/PREFICMD.

Example:

Exclude the L3 command for file 8 from prefetch processing.

```
ADARUN PREXFIL=8 ,PREFXCMD=L3
```

Multiple Exclusions

Specify multiple file numbers in multiple PREXFIL statements or in a list, separated by commas. Specify multiple commands in multiple PREFXCMD statements or in a list, separated by slashes (/). The maximum number of entries for each is 50.

If you use lists, they must have the same number of entries. There is a one-to-one correspondence, by position, between a file number in the PREXFIL list and a command in the PREFXCMD list.

Example:

Exclude the L3 command for file 8 and the L3 and L9 commands for file 12 from prefetch/multifetch processing.

```
ADARUN PREFXFIL=8,12,12,PREFXCMD=L3/L3/L9
```

OR

```
ADARUN PREFXFIL=8,PREFXCMD=L3
ADARUN PREFXFIL=12,PREFXCMD=L3
ADARUN PREFXFIL=12,PREFXCMD=L9
```

As the preceding example shows, you can specify the same file number several times, with a different command each time. Likewise, you can specify the same command code several times, with a different file number each time.

Eligible Commands and Command Pairs

The following commands can be excluded from prefetch/multifetch processing for a file:

- L1/L4
- L2/L5
- L3/L6
- L9

If either command in one of the pairs above is specified in a PREFXCMD statement, *both* commands are excluded. For example, if PREFXCMD=L1 is specified, both L1 and L4 are excluded.

Example:

Execute a user program with multifetch and allocate 20,000 bytes for the total prefetch buffer area. Exclude the L3 and L6 commands for files 8 and 12 from multifetch processing. Exclude *all* commands for file 7.

```
ADARUN PROG=USER,PREFETCH=YES,PREFTBL=20000
ADARUN PREFXFIL=8,12,PREFXCMD=L3/L3
ADARUN PREFXFIL=7,7,7,7,PREFXCMD=L1/L2/L3/L9
```

No PREFSBL value is specified, so the buffer length for a single prefetch sequence defaults to 3,000 bytes; up to 6 prefetch sequences can be handled.

PROGRAM : Program to Run

Parameter	Specify . . .	Possible Values	Default
PROGRAM	the program to be executed.	ADANUC USER utility-name node-name	USER

This parameter specifies what to execute: the Adabas nucleus, an Adabas utility, or a user-written program.

To start the Adabas nucleus, specify PROGRAM=ADANUC. See *Adabas Session Execution* for more information about executing an Adabas nucleus.

To execute a user-written program, specify PROGRAM=USER. *Linking Applications to Adabas* provides information about implementing user-written programs.

To execute an Adabas utility, specify one of the following names:

ADAACK	Address converter check utility
ADACDC	Changed data capture utility
ADACMP	Data compression/decompression utility
ADACNV	Database conversion/reversion utility
ADADBS	Database services utility
ADADCK	Data Storage check utility
ADADEF	Database definition
ADAFRM	Formatting utility
ADAICK	Inverted list check utility
ADAINV	Invert utility
ADALOD	Loader utility
ADAMER	ADAM statistics utility
ADAORD	Reorder utility
ADAPLP	Protection log print utility
ADAPRI	Maintenance print utility
ADARAI	Recovery Aid control utility
ADAREP	Database report utility
ADARES	Restart utility
ADASAV	Save/restore utility
ADASCR	Adabas Security utility
ADASEL	Audit trail utility
ADAULD	File unload utility
ADAVAl	Adabas validity check utility

Some of the special module names you might encounter include the following:

ADAREV	To start an Adabas Review hub
ADACOM	To set up an Adabas cluster environment
NETWRK	To set up an Entire Net-Work node

Refer to the documentation for Adabas Review or Entire Net-Work, respectively, for more information about these ADARUN programs.

Example 1:

Start the Adabas nucleus.

ADARUN PROGRAM=ADANUC

Example 2:

Execute a user-written program.

ADARUN PROGRAM=USER

Example 3:

Execute the ADACMP utility.

ADARUN PROGRAM=ADACMP

QBLKSIZE : Block Size for Sequential Datasets

Parameter	Specify . . .	Minimum	Maximum	Default
QBLKSIZE	the block size for sequential datasets used by ADAIOR.	0	262144	0 (see text)

QBLKSIZE specifies the block size, in bytes, for the sequential datasets used by ADAIOR, the Adabas I/O component. Use this parameter to override the default block sizes.



Warning:

If you choose to write tape files with large block sizes, for example for database backups, these files will not be transportable to systems where support for large blocks is not available. This might include a site being used as a backup facility for disaster recovery.

If QBLKSIZE=0 (the default), the following default block sizes are used:

Dataset Block Size

Tape the optimum selected for the device by the operating system

CKD-Disk track-size / k, where k = smallest value such that track-size / k <= 32760

FBA-Disk 32760

Under all operating systems except VSE/ESA, this block size may be provided in the JCL or dataset label.

Under BS2000, disk block sizes are always rounded up to the next integer multiple of 2048 bytes.

Example:

During this Adabas session, use a block size of 20,000 bytes for the Adabas sequential datasets.

```
ADARUN  PROG=ADANUC,QBLKSIZE=20000
```

READONLY : Read-Only Session Control

Parameter	Specify . . .	Possible Values	Default
READONLY	whether to permit database updates during this session.	YES NO	NO

READONLY determines whether the Adabas session is a read-only or a read/write session.

Value	Meaning
YES	All database update commands issued by this nucleus are rejected.
NO	The default setting. The session allows both reading and updating of the database.

Guidelines

- When executing the ADAREP (database report) utility, you can specify READONLY=YES to issue an access OPEN instead of an update OPEN.
- When using READONLY=YES, ensure that the Work dataset does not contain a pending autorestart; otherwise, the nucleus cannot start.
- Each nucleus being run must have its own Work dataset.
- When a nucleus is started with READONLY=YES, it is *not* possible to switch the nucleus to READONLY=NO mode later in the session. When a nucleus is started with READONLY=NO, it *can* be switched back and forth between the READONLY=YES and READONLY=NO modes.
- A database save performed while the nucleus is in read-only mode is equivalent to an offline save (which is taken while the nucleus is down). You can avoid shutting down the nucleus for an offline save by switching the nucleus into read-only mode before the save and back to update mode after the save.

Example:

Start this Adabas session as read-only.

```
ADARUN PROG=ADANUC,READONLY=YES
```

REVIEW : Adabas Review Control

Parameter	Specify . . .	Possible Values	Default
<u>REVIEW</u>	whether to run Adabas Review in local or hub mode specifying the hub ID, or at all.	NO <u>LOCAL</u> dbid	NO

Note:

REVIEW replaces the ADARUN parameter REVIEWHUBID introduced in Adabas version 6; however, REVIEWHUBID remains a synonym for REVIEW to ensure downward compatibility.

REVIEW controls the use of the Adabas Review product:

Value	Meaning
NO	The default setting. Adabas Review is not started.
<u>LOCAL</u>	Adabas Review 4.2 and above is started in local mode running as an extension to ADALOG. This is the only active Adabas Review setting available under VM/ESA. This setting is not valid for Adabas Review 4.1, which runs as an Adabas user exit 4. Instead, you must set ADARUN UEX4=RAOSEXIT and the CLOG, LOGGING, and LOGxx parameters specified in the Adabas Review version 4.1 documentation. In addition, you must apply ZAP OD41301 to Adabas Review version 4.1.3, the only supported level of Adabas Review 4.1.
dbid	Adabas Review is started in hub mode. The physical database ID that you specify for the hub identifies <ul style="list-style-type: none"> ● the hub (server) itself (with PROGRAM=ADAREV) that is being started; or ● from an Adabas nucleus (client), the hub that is the target for Adabas Review processing for that nucleus (with PROGRAM=ADANUC).

Dynamic Modification

The setting of the ADARUN PROG=ADANUC,REVIEW=dbid parameter can be changed dynamically using the REVIEWHUBID command from the operator console, the ADADBS OPERCOM REVIEWHUBID function, or the Modify Parameter function of Adabas Online System.

Example 1:

For the Adabas Review hub (server) installation, start hub 202.

```
ADARUN PROGRAM=ADAREV,REVIEW=202
```

Example 2:

For the Adabas Review (client) installation, start the Adabas nucleus that will log to Adabas Review hub 202.

```
ADARUN PROGRAM=ADANUC,REVIEW=202
```

SMGT : Error Handling and Message Buffering Facility

Parameter	Specify . . .	Possible Values	Default
<u>SMGT</u>	whether to enable the error handling and message buffering facility.	YES NO	NO

Value	Meaning
YES	ADARUN loads the module ADAMXI during initialization. If you want to use the message buffering functions, you must also specify the ADARUN MSGBUF parameter with a value greater than zero.
NO	(default) Error handling and message buffering is not enabled for the session.

This parameter can only be used when starting the nucleus. It cannot be used in utilities or user programs.

For more information about the error handling and message buffering facility, see the Adabas DBA Tasks documentation.

For information about temporarily turning the facility off and back on, see the SMGT operator command.

Example:

Activate the error handling and message buffering facility with a buffer size of 36 kilobytes when the Adabas nucleus is initialized.

```
ADARUN PROGRAM=ADANUC
ADARUN SMGT=YES
ADARUN MSGBUF=36
```

SORTCACHE : Sort Area Controller Caching

This parameter is valid for the operating environments OS/390, z/OS, VM/ESA, and z/VM only.

Parameter	Specify . . .	Possible Values	Default
<u>SORTCACHE</u>	whether to enable (YES) or disable (NO) controller caching for the Adabas sort dataset.	YES NO	YES

SORTCACHE controls caching for the sort area component of Adabas.

Value	Meaning
YES	The default setting. Sort blocks are preread and maintained as long as possible in the caching facilities of the disk controller.
NO	Sort blocks are not preread or stored.

Guidelines

- If sort blocks are highly scattered or the update frequency is low, controller caching can cause longer I/O response times.
- ADARUN caching control parameters also exist for the Associator (ASSOCACHE), Data Storage (DATACACHE), the sort area (WORKCACHE), and the temp area (TEMPCACHE).

Example:

Run the nucleus without controller caching for the sort dataset.

```
ADARUN PROG=ADANUC , SORTCACHE=NO
```

SPT : Activate Adabas Triggers and Stored Procedures

Parameter	Specify . . .	Possible Values	Default
<u>S</u> P <u>T</u>	whether to enable use of triggers and stored procedures.	YES NO	NO

When the Adabas nucleus starts and SPT=YES has been specified, it passes control to the Adabas trigger driver (ADATSP) to allow it to initialize.

This parameter can only be used when starting the nucleus and only in multiuser mode. It cannot be used for utilities or user programs and it cannot be used in single-user mode.

For more information, see the Adabas Triggers and Stored Procedures documentation.

Example:

Set Adabas to use the triggers and stored procedures functions.

```
ADARUN PROGRAM=ADANUC,MODE=MULTI,SPT=YES
```

SVC : SVC Number

This parameter applies to the operating environments OS/390, z/OS, and VSE/ESA only.

Parameter	Specify . . .	Possible Values	Default
<u>SVC</u>	the Adabas SVC number to be used for the session.	see text	45 (VSE) 249 (OS)

The SVC number is specified as an integer. It must correspond to the number used for the Adabas SVC at your installation.

The Adabas SVC is used to perform various Adabas internal functions under OS/390, z/OS, and VSE/ESA.

Valid SVC values are as follows:

OS/390 or 200-255

z/OS

VSE/ESA 45 is recommended; any free SVC value can be used. See the Adabas Installation documentation for information about finding free values for VSE.

Example:

Execute an Adabas session under OS/390 or z/OS using SVC 202 for the Adabas SVC.

```
ADARUN PROG=ADANUC,SVC=202
```

TAPEREL : Tape Handling Control (BS2000 Only)

Parameter	Specify . . .	Possible Values	Default
TAPEREL	end-of-file (EOF) handling for tapes and cartridges.	KEEPUNL KEEP NO RELEASE UNLOAD	KEEPUNL

Value Meaning

KEEPUNL (default) Retain the I/O unit assignment but unload the tape or cartridge.

KEEP Retain the I/O unit assignment without unloading.

NO Do not perform the RELEASE macro function.

RELEASE Do not keep the I/O unit assignment; do not unload the tape or cartridge.

UNLOAD Do not keep the I/O assignment; unload the tape/cartridge.

Example:

After the ADAULD utility is executed, the I/O assignment remains, and the tape/cartridge remains loaded.

```
ADARUN PROG=ADAULD,TAP=KEEP
```

TASKCTGY : Adabas Batch/TP Task Category Control

This parameter applies in BS2000 operating environments only.

Parameter	Specify . . .	Possible Values	Default
TASKCTGY	the Adabas task category.	BATCH TP	BATCH

BS2000 TP (interactive processing) environments favor TP-processing tasks over batch tasks. By default, Adabas has batch status. You can use the TASKCTGY parameter to assign TP-processing priority to the Adabas nucleus.

Value	Meaning
BATCH	(default) The nucleus retains batch status.
TP	Issues a TINF macro to force the Adabas nucleus to TP status.

Example:

Force the nucleus to TP-processing status.

```
ADARUN PROG=ADANUC, TASKCTGY=TP
```

TCPIP : TCP/IP Access Control (OS/390 or z/OS Only)

Parameter	Specify . . .	Possible Values	Default
<u>TCPIP</u>	whether to activate a direct TCP/IP link to the Adabas nucleus.	YES NO	NO

For UES-enabled databases, Adabas version 7 supports a direct TCP/IP link to the OS/390 or z/OS Adabas nucleus from web-based applications or from PC-based applications such as Natural for Windows.

See the Adabas Installation documentation for information about installing direct TCP/IP database access.

The TCPIP parameter is used to activate a direct TCP/IP link to the Adabas nucleus.

If TCPIP=YES, you can specify the universal resource locator (URL) required to activate the direct TCP/IP link in the ADARUN parameter TCPURL.

Example:

Run Adabas with a direct TCP/IP link to the nucleus.

```
ADARUN PROG=ADANUC,TCPIP=YES,TCPURL=HPS://STACKNAME:12345
```

TCPURL : TCP/IP Universal Resource Locator

Parameter	Specify . . .	Possible Values	Default
<u>TCPURL</u>	the universal resource locator (URL) for the TCP/IP link.	(see text)	none

If TCPIP=YES, you can specify the information required to activate the direct TCP/IP link to the Adabas nucleus. The parameter value is a 20-byte address that conforms to the RFC specification for universal resource locators (URLs):

TCPURL=api-name://stackid:port-number

where

api-name is a 1-3 character value identifying the application programming interface (API) to use. Both APIs for the IBM TCP/IP stack (HPS, OES) and the API for the Interlink stack (ILK) are currently supported.

stackid is a 1-8 character value identifying the stack to use. For the HPS API, this is the name of the TCP/IP started task. For the OES API, no value is needed. For the ILK API, this is the subsystem identifier.

port-number is a 1-5 character number in decimal notation.

Optionally, you can specify this and additional URLs using the operator command TCPIP.

Example:

Run Adabas with a direct TCP/IP link to the nucleus.

ADARUN PROG=ADANUC,TCPIP=YES,TCPURL=HPS://STACKNAME:12213

TEMPCACHE : Temp Area Controller Caching

This parameter is valid for the operating environments OS/390, z/OS, VM/ESA, and z/VM only.

Parameter	Specify . . .	Possible Values	Default
TEMPCACHE	whether to enable (YES) or disable (NO) controller caching for the Adabas temp dataset.	YES NO	YES

Value	Meaning
YES	The default setting. Temp blocks are preread and maintained as long as possible in the caching facilities of the disk controller.
NO	Temp blocks are not preread or stored.

Guidelines

- If temp blocks are highly scattered or the update frequency is low, controller caching can cause longer I/O response times.
- ADARUN caching control parameters also exist for the Associator (ASSOCACHE), Data Storage (DATACACHE), the work area (WORKCACHE), and the sort area (SORTCACHE).

Example:

Run the nucleus without controller caching for the temp dataset.

```
ADARUN PROG=ADANUC,TEMPCACHE=NO
```

TFLUSH : Synchronous Buffer Flush Duration

Parameter	Specify . . .	Minimum	Maximum	Default
TFLUSH	the time allowed for a synchronous buffer flush.	1	16 777 215	1

The TFLUSH value specifies the time allowed for a synchronous buffer flush, in seconds.

If the LFIOP parameter specifies a non-zero value, the TFLUSH value has no effect because asynchronous buffer flush control is active.

If LFIOP=0 (the default) and if TFLUSH is not specified or has the value 1, one second is allowed for each synchronous buffer flush.

Example:

During this Adabas session, 6 seconds are allowed for each synchronous buffer flush.

```
ADARUN PROG=ADANUC,TFLUSH=6,LFIOP=0, . . .
```

TLSCMD : Time Limit for S1, S2, and S4 Complex Searches

Parameter	Specify . . .	Minimum	Maximum	Default
TLSCMD	the time limit for executing a single S1, S2, or S4 command with a complex search criterion.	1	16 777 215	300

The TLSCMD parameter specifies the maximum time (in units of 1.048576 seconds) to be used to process a single Adabas S1, S2, or S4 command with a complex search criterion.

If the estimated time to process a command exceeds the TLSCMD time limit, the command is not processed and the user who issued the command receives a nonzero response code.

If the estimated time to process a command does not exceed the time limit, the command is processed to completion, whether the actual time required exceeds the time limit specified by the TLSCMD or not.

If the TLSCMD setting exceeds the maximum time permitted by the Adabas nucleus (currently 900), the maximum time is used.

Example:

During this Adabas session, the time limit for an S1, S2, or S4 command with a complex search criterion is about 50 minutes.

```
ADARUN PROG=ADANUC, TLSCMD=3000
```

Overriding the Parameter Setting

You can override this parameter setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.

TNAA : Nonactivity Time Limit for Access-Only Users

Parameter	Specify . . .	Minimum	Maximum	Default
TNAA	the nonactivity time limit for access-only users.	1	16 777 215	900

The TNAA parameter specifies the maximum elapsed time (in units of 1.048576 seconds) that an access-only user may be considered active without issuing an Adabas command.

If an access-only user exceeds the nonactivity time limit, action is taken as described in *Adabas Timeout Feature*.

If the SMPID parameter specifies a value greater than zero (that is, a multiprocessing nucleus), you may have to specify a larger TNAA value than for a normal nucleus. The chances of data contention are greater, and therefore more time should be allowed.

Example:

During this Adabas session, the nonactivity time limit for access-only users is about 30 minutes.

```
ADARUN PROG=ADANUC, TNAA=1800
```

Overriding the Parameter Setting

You can override this parameter setting for an individual user or for all users:

- Override the setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.
- Override the setting for *all* users with an Adabas TNAA operator command or with the OPERCOM function of the ADADBS utility (described in the Adabas Utilities documentation).

TNAE : Nonactivity Time Limit for ET Logic Users

Parameter	Specify . . .	Minimum	Maximum	Default
TNAE	the nonactivity time limit for ET logic users.	1	16 777 215	900

The TNAE parameter specifies the maximum elapsed time (in units of 1.048576 seconds) an ET Logic user may be considered active without issuing an Adabas command.

If an ET logic user exceeds the nonactivity time limit, action is taken as described in *Adabas Timeout Feature*.

If the SMPID parameter specifies a value greater than zero (that is, a multiprocessing nucleus), you may have to specify a larger TNAE value than for a normal nucleus. The chances of data contention are greater, and therefore more time should be allowed.

Example:

During this Adabas session, the nonactivity time limit for ET logic users is about 30 minutes.

```
ADARUN PROG=ADANUC ,TNAE=1800
```

Overriding the Parameter Setting

You can override this parameter setting for an individual user or for all users:

- Override the setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.
- Override the setting for all users with an Adabas TNAE operator command or with the OPERCOM function of the ADADBS utility (described in the Adabas Utilities documentation).

TNAX : Nonactivity Time Limit for Exclusive Control Users

Parameter	Specify . . .	Minimum	Maximum	Default
TNAX	the nonactivity time limit for exclusive control users.	1	16 777 215	900

The TNAX parameter specifies the maximum elapsed time (in units of 1.048576 seconds) an exclusive control user (EXU) may be considered active without issuing an Adabas command.

If an exclusive control user exceeds the nonactivity time limit, action is taken as described in *Adabas Timeout Feature*.

If the SMPID parameter specifies a value greater than zero (that is, a multiprocessing nucleus), you may have to specify a larger TNAX value than for a normal nucleus. The chances of data contention are greater, and therefore more time should be allowed.

Example:

During this Adabas session, the nonactivity time limit for exclusive control users is about 30 minutes.

```
ADARUN PROG=ADANUC, TNAX=1800
```

Overriding the Parameter Setting

You can override this parameter setting for an individual user or for all users:

- Override the setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.
- Override the setting for all users with an Adabas TNAX operator command or with the OPERCOM function of the ADADBS utility (described in the Adabas Utilities documentation).

TT : Transaction Time Limit

Parameter	Specify . . .	Minimum	Maximum	Default
TT	the transaction time limit for ET logic users.	1	16 777 215	900

The TT parameter specifies the maximum elapsed time (in units of 1.048576 seconds) permitted for a logical transaction issued by an ET logic user.

The time measurement for a logical transaction starts when the first command that results in a record being placed in hold status is issued. It ends when an ET, BT, or CL command is issued.

If the transaction time limit is exceeded, action is taken as described in *Adabas Timeout Feature* .

The value specified for TT directly influences the required size of the Adabas parameter LP (length of the data protection area).

Software AG recommends that you set the TT parameter lower than the TNAE parameter.

Example:

During this Adabas session, the transaction time limit for ET logic users is about 5 minutes (300 seconds).

```
ADARUN PROG=ADANUC,TT=300
```

Overriding the Parameter Setting

You can override this parameter setting for an individual user or for all users:

- Override the setting for an individual user by specifying a different value in the Adabas control block for an OP command. See the discussion of the OP command in the Adabas Command Reference documentation.
- Override the setting for all users with an Adabas TT operator command or with the OPERCOM function of the ADADBS utility (described in the Adabas Utilities documentation).

UEXn : User Exit

Parameter	Specify . . .	Possible Values	Default
UEXn	the user exit and the user routine to be given control.	see text	none

This parameter is used in conjunction with the user exit facility. It specifies one or more optional user exits and their names in the following format:

UEXn=exit-name

where

n is a digit in the range 1-12 inclusive

exit-name is the name of a user routine that gets control at the user exit; the name can be up to 8 characters long.

Note:

User exit 2 and user exit 12 are mutually exclusive; if you specify one, you may not specify the other.

The specified user exit routine(s) must be loadable at execution time.

The User Exits documentation describes the purpose and function of each user exit as well as the calling sequence.

Example 1:

During this Adabas session, give control to the user routine "SECURE" at user exit 1.

```
ADARUN PROG=ADANUC,UEX1=SECURE
```

Example 2:

During this Adabas session, give control to the user routine "SUBR2" at user exit 2; give control to the user routine "SUBR4" at user exit 4.

```
ADARUN PROG=ADANUC,UEX2=SUBR2,UEX4=SUBR4
```

Example 3:

Execute the ADACMP utility and give control to the user routine "SUBR6" at user exit 6.

```
ADARUN PROG=ADACMP,UEX6=SUBR6
```

UTIONLY : Utilities-Only Session Control

Parameter	Specify ...	Possible Values	Default
UTIONLY	whether to restrict the session to Adabas utilities.	YES NO	NO

Use this parameter to restrict an Adabas session to the execution of Adabas utilities.

Value	Meaning
YES	Any attempt to execute a program other than an Adabas utility is rejected.
NO	(default) The session is available for both utilities and user programs.

If you specify UTIONLY=YES to reserve the session for a utility program, you can use the operator command UTIONLY=NO to make the nucleus available for user programs again when the utility is completed.

Example:

During this Adabas session, only utilities can be executed.

```
ADARUN PROG=ADANUC,UTIONLY=YES
```

VISTA : Adabas Vista Control

Parameter	Specify . . .	Possible Values	Default
VISTA	whether to enable support for Adabas Vista.	YES NO	NO

The VISTA parameter enables the nucleus to support Adabas Vista processing. If Adabas Vista is not installed on your system, specifying this parameter invokes the AVILOOK analysis tool delivered with Adabas.

Example:

Run Adabas with Adabas Vista.

```
ADARUN PROG=ADANUC,VISTA=YES
```

WORKCACHE : Work Area Controller Caching

This parameter is valid for the operating environments OS/390, z/OS, VM/ESA, and z/VM only.

Parameter	Specify . . .	Possible Values	Default
<u>WORKCACHE</u>	whether to enable (YES) or disable (NO) controller caching for the Adabas Work dataset.	YES NO	YES

Value	Meaning
YES	The default setting. Work blocks are preread and maintained as long as possible in the caching facilities of the disk controller.
NO	Work blocks are not preread or stored.

Guidelines

- If Work blocks are highly scattered or the update frequency is low, controller caching can cause longer I/O response times.
- ADARUN caching control parameters also exist for the Associator (ASSOCACHE), Data Storage (DATACACHE), the sort area (WORKCACHE), and the temp area (TEMPCACHE).

Example:

Run the nucleus without controller caching for the Work dataset.

```
ADARUN PROG=ADANUC,WORKCACHE=NO
```

Operator Commands

Adabas operator commands are entered during an Adabas session or during utility operation to

- terminate an Adabas or user session;
- display nucleus or utility information;
- log commands into CLOG;
- change Adabas operating parameters or conditions.

In this chapter, the commands are listed alphabetically. One command, DSTAT, is listed twice: once as a command for displaying nucleus status, and separately as a command to display current Adabas utility operating status.

- For Adabas Caching Facility operator commands, see the Adabas Caching Facility documentation.
- For Adabas Parallel Services operator commands, see the Adabas Parallel Services documentation.
- For operator commands in a sysplex cluster environment, see the Adabas Cluster Services documentation.

This chapter covers the following topics:

- Entering Operator Commands
- Operator Commands
- SMGT Display Output Samples

Entering Operator Commands

BS2000 Systems

In BS2000 environments, enter each command at the operator console by addressing the Adabas nucleus with its task sequence number (TSN) in the following form:

```
//INTR TSN, command
```

For testing purposes, the nucleus may be run as a dialogue process. The nucleus may be interrupted by pressing the K2 key, after which the prompt "/" appears. Now an operator command can be sent to the nucleus in the following form:

```
//INTR command
```

Note:

In the dialogue mode, the nucleus stops as long as the INTR message is not sent back. The resume statement /RESUME causes the nucleus to resume where it was interrupted when no operator command is issued.

Operator commands are processed by a STXIT routine.

OS/390 or z/OS Systems

To enter operator commands in OS environments, use the OS MODIFY (F) command as shown below:

```
F jobname,command
```

where *jobname* is the name specified by the EXEC job control statement (usually ADARUN).

VM/ESA or z/VM Systems

In VM/ESA environments, use the CP SEND command to enter operator commands from either the console of the virtual machine or from a secondary console of a disconnected virtual machine.

Example:

```
CP SEND USER04 ADAEND
```

- issued at the DBA virtual machine console sends the ADAEND command to the USER04 virtual machine, providing that the CP directory for the USER04 virtual machine contains a CONSOLE statement enabling the DBA console as a secondary console.

VSE/ESA Systems

To enter operator commands in VSE/ESA environments, use the following steps:

1. Enter an MSG command for the VSE/ESA partition in which Adabas is executing.

Example:

```
MSG Fn
```

When ready for communication, Adabas will respond with Adabas message ADAI29. In cases where the commands ADAEND, CANCEL, and HALT cause Adabas to end the nucleus session, no outstanding reply is presented so that an orderly shutdown can occur without the need for operator intervention.

2. Enter the desired Adabas operator command or commands.

More than one command can be entered in a session. You can also enter a command followed directly by a slash (/) to end any further operator communications until the next MSG command by VSE/ESA.

3. When all desired commands have been entered, close the operator communication session by entering a null command (EOB).
4. To enter any more commands after the session is closed, repeat this entire procedure.

Operator Commands

The Adabas operator commands are grouped as follows:

- Nucleus Commands
- Adabas Utility Operator Command DSTAT
- SMGT Operator Command

Nucleus Commands

The following operator commands can be entered from the console to monitor and control Adabas nucleus operation.

Note:

The DUMP command, which stopped nucleus operation and issued a dump, is no longer supported. To stop operation, use an operating system-dependent command such as an OS Cancel.

ADAEND [GLOBAL]

Terminate Adabas session normally. No new users are accepted after this command has been issued. ET logic updating is continued until the end of the current logical transaction for each user. After all activity has been completed as described above, the Adabas session is terminated.

In nucleus cluster environments, the GLOBAL option can be used to terminate the Adabas session in all active cluster nuclei.

{ ADDCLOG | ADDPLOG }=*dsname*

Dynamically add a command log (CLOG) or protection log (PLOG) dataset. The dataset must already have been specified in the session startup JCL and must already have been formatted using ADAFRM. A maximum of 8 CLOG and 8 PLOG datasets may be used.

ALOCKF=*n*

Lock a file in advance to ensure that an EXU, EXF, or UTI user will have exclusive control of the specified file. The advance-lock prevents new transactions from using the file. Once all current users have stopped using the file, the exclusive-control user has the lock. Until then, the exclusive-control user must wait.

To remove the advance lock without running the utility, see the RALOCKF command.

This command is not available in single user mode or for a read-only nucleus.

AOSLOG={YES | NO}

Activate/deactive logging of certain Adabas calls that modify the nucleus to DD/PRINT. These calls are issued either by ADADBS OPERCOM or Adabas Online System. Read/display calls are not logged.

ASYTVS={YES | NO}

Activate/deactive asynchronous flushing of buffers based on volume serial number.

CANCEL [,GLOBAL]

Cancel Adabas session immediately. All command processing is immediately suspended. A pending autorestart will be in effect which in turn will cause the autorestart routine to be executed during the initialization of the next Adabas session.

In nucleus cluster environments, the GLOBAL option can be used to cancel the Adabas session in all active cluster nuclei.

CLOGMRG={YES | NO}

Dynamically modify the setting of the ADARUN CLOGMRG parameter.

The CLOGMRG command is only valid in cluster environments. It is global by definition and affects all nuclei in the cluster.

CLUFREEUSER=[,TNA= *max-time*][,UID=*userid*][,FORCE][,GLOBAL]

Note:

The CLUFREEUSER command is only valid in cluster environments. It can be issued against the local nucleus only or, with the GLOBAL option, against all active and inactive nuclei in the cluster.

Delete leftover user table elements (UTES) in common storage that are no longer associated with user queue elements (UQEs) in a nucleus where

TNA is a decimal number specifying the timeout value in seconds. UTEs that are not used during the time specified may be deleted if other conditions are fulfilled. If TNA= is not specified, UTEs may be deleted without regard to their recent use.

UID is a character string or hexadecimal byte string as follows:

cccccccc where the argument is 1-8 letters, digits, or embedded '-' signs without surrounding apostrophes.

'cccccccc' where the argument is 1-8 characters with surrounding apostrophes.

X'xxxxxxxxxxxxxxxx' where the argument is an even number of 2-16 hexadecimal digits enclosed by X' '.

A character string must be enclosed in apostrophes if it contains characters other than letter, digits, or embedded '-' signs. If a specified character string is less than 8 characters long, it is implicitly padded with blanks. If a specified hexadecimal string is shorter than 16 hexadecimal digits, it is implicitly padded with binary zeros. If the UID string ends with a '*' (or X'5C'), the preceding characters are treated as a user ID prefix: only the prefix is needed for an argument match. If the last 8 bytes of a user's 28-byte communication ID match a specific user ID or user ID prefix, that user's UTE may be deleted if other conditions are fulfilled. If UID= not specified, UTEs may be deleted regardless of their user IDs.

FORCE Leftover UTEs are to be deleted even if the users are due a response code 9, subcode 20. If FORCE is not specified, such UTEs are not deleted. Before using the FORCE parameter, ensure that the users owning the UTEs to be deleted will not expect any of their transactions to remain open.

GLOBAL Leftover UTEs throughout the Adabas cluster are to be deleted if they are no longer associated with UQEs and are eligible according to the other specified parameters. Additionally and subject to the other rules, leftover UTEs are deleted if their assigned nuclei have terminated since their last use. If GLOBAL is not specified, only UTEs assigned to the local nucleus and used since the nucleus start are eligible for deletion.

CT=timeout-limit

Dynamically override the ADARUN CT parameter value; that is, the maximum number of seconds that can elapse from the time an Adabas command has been completed until the results are returned to the user through interregion communication (which depends on the particular operating system being used). The minimum setting is 1; the maximum is 16777215.

In nucleus cluster environments, the CT command is global by definition and affects all nuclei in the cluster.

DAUQ

Display the user queue elements of those users who have executed at least one Adabas command within the last 15 minutes.

DCQ

Display all posted command queue elements (CQEs). The DCQ command displays each CQE's user ID, job name, and buffer length.

DDIB

Display data integrity block (DIB). This block contains entries indicating which Adabas utilities are active and the resources being used by each utility.

DDSF

Display Delta Save Facility status. The DDSF command is only available if the Adabas nucleus is run with the parameter ADARUN DSF=YES.

{ DELCLOG | DELPLOG }= *dsname*

Dynamically delete a command log (CLOG) or protection log (PLOG) dataset. The current number of CLOGs/PLOGs for the session must remain greater than 1 or an error is returned.

DELUF=*file-number*

Delete all user queue elements for the specified file. The DELUF command corresponds to the ADADBS OPERCOM STOPF=*file-number*,PURGE function.

DELUI=*timeout-limit*

Delete all user queue elements that remained inactive for the specified amount of time (specified in seconds). The DELUI command corresponds to the ADADBS OPERCOM STOPI=time,PURGE function.

```
DFILES={ n | n1-nx | n1,..., n5 }
```

Display the number of users currently accessing, updating, or controlling either file "n", a range of files "n1 - nx" , or files "n1,...,n5". A maximum of five files can be specified. Users are displayed by job name and Adabas-assigned user ID, and listed by file.

```
DFILUSE=nnn
```

Display the count of total commands processed so far for the specified file during the current session. The count is displayed in the nucleus message ADAN33.

```
DHQ
```

Display up to five hold queue elements.

```
DHQA
```

Display up to 1000 hold queue elements.

```
DLOCKF
```

Display locked files.

```
DNC
```

Display the number of posted command queue elements.

```
DNFV
```

Display nucleus file variables; that is, information about current file use.

DNH

Display the number of ISNs currently in the hold queue.

DNU

Display the number of current users.

DONLSTAT

Display the status of each active reorder or invert online process together with the process ID.

DPARM

Display the Adabas session parameters currently in effect.

DPPT

Display the parallel participant table (PPT); that is, display the nucleus's own PPT block.

DRES

Display the allocated pool space and the highest use level ('high water mark') reached so far during the current session by record count and by percent for the following resources:

- attached buffers (AB) - current allocation not supported
- command queue (CQ)
- format pool (FP)
- hold queue (HQ)
- pool for the table of ISNs (TBI)
- pool for the table of sequential commands (TBQ or TBLES)
- user queue (UQ)

- unique descriptor pool (DUQPOOL)
- security pool
- user queue file list pool
- work pool (WP)
- pool for global transaction IDs (XIDs; nonzero only with Adabas Transaction Manager)
- cluster block update "redo" pool (nonzero only with Adabas Cluster Services)

The actual values are displayed in nucleus message ADAN28 described in the Adabas Messages and Codes documentation.

DSTAT

Display the current Adabas nucleus status.

DTH

Display thread status.

DUQ

Display up to five active and inactive user queue elements.

DUQA

Display up to 100 user queue elements.

DUQE=X 'userid'

Display the user queue element for the specified Adabas-assigned user ID. The user ID must be entered in hexadecimal format as follows:

DUQE=X 'A3CF2'

Do not enter a job name in place of the user ID.

DUUQE

Display utility user queue elements.

```
FEOFCL [,GLOBAL]
```

Close the current dual or multiple command log and switch to the another command log. This command is valid only if dual or multiple command logging is in effect.

In nucleus cluster environments, the GLOBAL option can be used to close and switch dual or multiple command logs in all active cluster nuclei.

```
FEOFPL [,GLOBAL]
```

Close the current dual or multiple data protection log and switch to the another protection log. This command is valid only if dual or multiple data protection logging is in effect.

In nucleus cluster environments, the GLOBAL option can be used to close and switch dual or multiple protection logs in all active cluster nuclei.

```
FMXIO=nn
```

Dynamically modify the setting of the ADARUN FMXIO parameter.

```
HALT [,GLOBAL]
```

Perform a BT (back out transaction) for each active ET logic user's session, then terminate the Adabas session. No dumps are produced by HALT.

In nucleus cluster environments, the GLOBAL option can be used to perform a BT for all active ET logic user sessions and terminate the Adabas session in all active cluster nuclei.

```
LOCKF=n
```

Lock the specified file. No use of the specified file is possible at any security level.

```
LOCKU=n
```

Lock the specified file for all non-utility users. Adabas utilities can use the specified file normally.

LOCKX=*n*

Lock the specified file for all users except EXU or EXF users. EXU and EXF users can use the file normally. The lock is released automatically when an EXU user issues an OP command.

LOGGING

Begin command logging.

LOGCB

Begin logging of the Adabas control block for each command logged.

LOGFB

Begin logging of the Adabas format buffer for each command logged.

LOGIB

Begin logging of Adabas ISN buffer for each command logged.

LOGIO

Begin logging of Adabas I/O activity for each command logged.

LOGRB

Begin logging of the Adabas record buffer for each command logged.

LOGSB

Begin logging of the Adabas search buffer for each command logged.

LOGUX

Begin logging of user exit B data for inclusion in the CLOG record. This command is only valid when CLOGLAYOUT=5.

LOGVB

Begin logging of the Adabas value buffer for each command logged.

NOLOGGING

Stop or prevent command logging.

NOLOGCB

Stop or prevent logging of the Adabas control block.

NOLOGFB

Stop or prevent logging of the Adabas format buffer.

NOLOGIB

Stop or prevent logging of the Adabas ISN buffer.

NOLOGIO

Stop or prevent logging of Adabas I/O activity.

NOLOGRB

Stop or prevent logging of the Adabas record buffer.

```
NOLOGSB
```

Stop or prevent logging of the Adabas search buffer.

```
NOLOGUX
```

Stop logging of user exit B data for inclusion in the CLOG record. This command is only valid when CLOGLAYOUT=5.

```
NOLOGVB
```

Stop or prevent logging of the Adabas value buffer.

```
ONLRESUME=X'identifier'
```

Resume a previously suspended online reorder or invert process.

```
ONLSTOP=X'identifier'
```

Stop an online reorder or invert process cleanly. The process continues up to its next interrupt point in order to produce a consistent state, and then terminates after performing all necessary cleanup.

```
ONLSUSPEND=X'identifier'
```

Suspend an online reorder or invert process. The process continues up to its next interrupt point in order to produce a consistent state, performs a command throwback, and enters a state where it cannot be selected for processing. This command is useful if the online process is consuming too much of the nucleus resources.

```
RALOCKF=n
```

Remove the advance lock on the specified file (see ALOCKF command) without running the utility.

```
RALOCKFA
```

Remove the advance lock on all files for which it has been set (see ALOCKF command) without running the utility.

RDUMPST

Terminate online dump status. This command is normally used if online execution of the ADASAV utility has terminated abnormally.

READONLY={YES | NO}

Switches READONLY status on or off.

RESUME

Resume normal update transaction processing for the nucleus (or nuclei in a cluster environment) after suspension of transaction processing using ADADBS.

REVIEW={ NO | LOCAL | *hub-id* }

Deactivate Adabas Review; change from hub mode to local mode; specify or change the Adabas Review hub with which a nucleus communicates.

STOPF=*file-number*

Stop users who are using the specified file.

This command does not stop EXF or UTI users.

STOPI=*time*

Stop users who have not been executing during a given period of time (in seconds).

This command does not stop EXF or UTI users.

STOPU= $\left\{ \begin{array}{l} X *userid * \\ job-name \end{array} \right\}$

Note:

The `STOPU=X'userid'` command is not allowed for online reorder or invert processes. See the `ONLSTOP=X'identifier'` command instead.

Stop the user with the Adabas-assigned user ID (in the form shown in the display commands), or stop all users found with the job "job-name".

`STOPU/STOPF/STOPI` clear inactive or timed-out users and deletes the user queue element (UQE) for each identified user. If the program/user is an ET logic user, is not in ET status, and has not been stopped before one of these commands (`STOPU`, `STOPF`, or `STOPI`) is issued, Adabas backs out all updates made by the transaction to this point and releases all held records. If the transaction continues, only those changes following the `STOPU`, `STOPF`, or `STOPI` command are completed.

Note:

Before stopping a user, ensure that the user is in fact no longer active. If the stopped user returns (by sending a command), no indication is given about the transaction backout.

The user ID must be specified in hexadecimal format:

```
STOPU=X'A3CF2'
```



SYNCC

Force synchronization for all ET users. The nucleus waits for all ET users to reach ET status.



TCPIP={ OPEN=*url* | CLOSE=*url* | CLOSE }

Open or close a direct TCP/IP link to the Adabas nucleus or close all TCP/IP links when no URL is specified.

This command is only possible when the ADARUN parameter TCPIP is set to YES and all conditions for that setting have been met. This command can be used to close the URL set in the ADARUN TCPURL parameter, or to open/close additional TCP/IP links.

You must identify the universal resource locator (URL) for the TCP/IP link you want to open or close. The URL is a 20-byte address that conforms to the RFC specification for URLs:

api-name://stackid:port-number

where

api-name is a 1-3 character value identifying the application programming interface (API) to use. Both APIs for the IBM TCP/IP stack (HPS, OES) and the API for the Interlink stack (ILK) are currently supported.

stackid is a 1-8 character value identifying the stack to use:

- for the HPS API, this is the name of the TCP/IP started task.
- for the OES API, no value is needed.
- for the ILK API, this is the subsystem identifier.

port-number is a 1-5 character number in decimal notation.

Examples:

```
TCPIP=OPEN=ILK://ILZ5:1234
```

```
TCPIP=CLOSE=ILK://ILZ5:1234
```

To close all open URLs:

```
TCPIP=CLOSE
```



TNAA= n

Set non-activity time limit for access-only users. This value must be greater than zero, and replaces the value set by the TNAA ADARUN parameter.

In nucleus cluster environments, the TNAA command is global by definition and affects all nuclei in the cluster.



TNAE= n

Set non-activity time limit for ET logic users. This value must be greater than zero, and replaces the value set by the TNAE ADARUN parameter.

In nucleus cluster environments, the TNAE command is global by definition and affects all nuclei in the cluster.



TNAX= n

Set non-activity time limit for exclusive control users. This value must be greater than zero, and replaces the value set by the TNAX ADARUN parameter.

In nucleus cluster environments, the TNAX command is global by definition and affects all nuclei in the cluster.



TT=*n*

Set transaction time limit for ET logic users. This value must be greater than zero, and replaces the value set by the TT ADARUN parameter.

In nucleus cluster environments, the TT command is global by definition and affects all nuclei in the cluster.



UNLOCKF=*n*

Unlock the specified file. File usage is restored to its pre-locked status.



UNLOCKU=*n*

Unlock the specified file that was previously locked for all non-utility users. File usage is restored to its pre-locked status.



UNLOCKX=*n*

Unlock the specified file that was previously locked for non-exclusive control users. File usage is restored to its pre-locked status.



UTIONLY={YES | NO}

Switch UTIONLY status on or off. The default is NO.

Adabas Utility Operator Command DSTAT



DSTAT

Display the current ADALOD, ADAORD, ADARES, ADASAV, ADAULD, or ADAVAL operating status. The following are examples of the output that results when DSTAT is entered during specific Adabas utility operations:

Note:

ADAORD, ADARES, ADAULD, and ADAVAL do not accept operator commands if they run in MODE=SINGLE.

Example 1: ADALOD

```
ADAU00 dbid OPERATOR TYPE-IN : DSTAT ADAU22 dbid LOADING DATA STORAGE. REC-NO=3599
ADAU00 dbid OPERATOR TYPE-IN : DSTAT ADAU21 dbid SORTING/LOADING DESCRIPTOR AA
```

Example 2: ADAORD

```
ADAU08 dbid OPERATOR TYPE-IN : DSTAT ADAU26 dbid UNLOADING INDEX. FILE=1
ADAU08 dbid OPERATOR TYPE-IN : DSTAT ADAU25 dbid UNLOADING DATASTORAGE. FILE=2
ADAU08 dbid OPERATOR TYPE-IN : DSTAT ADAU29 dbid LOADING DATASTORAGE. FILE=2
```

Example 3: ADARES

```
ADAU08 dbid OPERATOR TYPE-IN : DSTAT REGENERATE INPUT VOLUME = BMC002, PLOG-NUM = 12
FROMBLK =          1, FROMTIME = 1996-04-10 11:27:56
TOBLK =           1, TOTIME   = 1996-04-10 11:27:56
```

Example 4: ADASAV

```
ADAU08 dbid OPERATOR TYPE-IN : DSTAT ADAU92 dbid          STILL INITIALIZING
ADAU08 dbid OPERATOR TYPE-IN : DSTAT ADAU10 dbid 435 BLOCKS OUT OF 465 SAVED
ADAU08 dbid OPERATOR TYPE-IN : DSTAT ADAU11 dbid 342 BLOCKS OUT OF 451 RESTORED
```

Example 5: ADAULD

```
ADAU08 dbid OPERATOR TYPE-IN : DSTAT ADAU67 dbid UNLOADING FILE=17, RECNO=2875
```

SMGT Operator Command

The Adabas error handling and message buffering facility uses a single operator command, SMGT, followed by a comma and one or more operands:

```
SMGT, operand,[ operand ],...
```

Operands may require that you enter one or more of the following variables:

exit-code	The code that identifies an exit routine to an SMGT operator command is one of the following: <ul style="list-style-type: none"> UEXn where "n" is a single-digit user exit number UXnn where "nn" is a double-digit user exit number HXnn where "nn" is a hyperexit number (single-digit numbers are preceded by a zero; e.g., HX02) CX00 the Adabas Caching Facility user exit SX00 the PIN routine user exit
start, end	The range of addresses for the SNAP operand where "start" is the hexadecimal address where the SNAP dump begins and "end" is the hexadecimal address where it ends.
module-name	The name of the module. For the XLOAD operand, it is the name of the exit routine module to be loaded; for ADDPIN and DELPIN operands, it is the name of the PIN module to be added or deleted, respectively.
pin-number	When a PIN is added, it is assigned a number that can be found using the DISPLAY=PINS operand. This pin number is used for the ACTPIN and DEACTPIN operands to identify the PIN routine to be activated or deactivated, respectively.

Acceptable abbreviations for operands are provided in parentheses; default values for operands are underlined.

Operands for SMGT

```
SMGT,{ ABNORMALTERM | MSGBUF } = { ON | OFF }
```

The ABNORMALTERM (ABN) operand determines whether the error handling and message buffering facility will handle abnormal termination errors.

The MSGBUF (MSG) operand temporarily deactivates (OFF) or reactivates (ON) message buffering.

```
SMGT,{ ACTPIN | DEACTPIN } =pin-number
```

The ACTPIN (ACT) operand is used to activate an individual PIN routine or to reactivate it after it has been temporarily deactivated.

The DEACTPIN (DEACT) operand is used to deactivate an individual PIN routine.

```
SMGT,{ ADDPIN | DELPIN } = module-name
```

The ADDPIN (ADD) operand adds PIN routines to the error handling facility. The PIN routine module indicated is loaded and the PINs that are found in it are added to the facility. When a PIN is added, it is assigned a number which can be found using the DISPLAY=PINS operand.

The DELPIN (DEL) operand deletes a PIN module and all the PINs it contains from the facility.

```
SMGT,DISPLAY = { ALL | EXITS | MSGBUF | PINS | SUMMARY | LAST }
```

The DISPLAY (D) operand writes status and history information about the error handling and message buffering facility to the job log and to DDPRINT:

ALL	(the default) displays all reports
EXITS	displays current user exit, hyperexit, and other exit status
MSGBUF	displays message buffering status
PINS	displays PIN routine status
SUMMARY	displays only the summary information from ALL
LAST	displays details of the most recent recovery action

Sample output for the various DISPLAY values is provided in *SMGT Display Output Samples* .

```
SMGT,DUMP = { ON | OFF }
```

The DUMP operand determines whether a full system dump is taken for the Adabas nucleus in the event of an error. The default (OFF) means that only a snap dump is taken.

Note:

To use this command successfully under OS/390 or z/OS, the dataset ADASNAP must be provided in the Adabas start-up JCL. See the chapter Adabas Session Execution for more information.

```
SMGT,{ ON | OFF }
```

The OFF operand deactivates the error handling and message buffering facility temporarily; the ON operand reactivates it.

When SMGT=OFF:

- the only valid SMGT commands are ON/OFF and DISPLAY. All other SMGT commands are rejected until SMGT=ON is issued.

- all other functions of the error handling facility are disabled. PIN modules cannot be invoked. Any user exits marked NOTCRITICAL are treated as CRITICAL; that is, the nucleus terminates abnormally if an error occurs in the exit.

```
SMGT,SNAP [= (start, end) ]
```

The SNAP operand displays a formatted dump of the nucleus without error diagnostics. If SNAP is specified without any additional parameters, the whole nucleus is displayed; if hexadecimal addresses are specified for start and end, the SNAP dump is displayed only for that range of addresses.

The SNAP command can degrade system performance as long as it is active.

Note:

To use this command successfully under OS/390 or z/OS, the dataset ADASNAP must be provided in the Adabas start-up JCL. See the chapter Adabas Session Execution for more information.

```
SMGT,{ XACTIVATE | XDEACTIVATE }=exit-code
```

The XACTIVATE (XA) operand activates a loaded exit module; the XDEACTIVATE (XD) operand deactivates an active exit module.

```
SMGT,{ XNOTCRITICAL | XCRITICAL }=exit-code
```

The XNOTCRITICAL (XN) operand changes the status of an exit from the default "critical" to "noncritical" for the functioning of the Adabas nucleus. Any abnormal termination or program check in a "noncritical" exit results in the exit being automatically deactivated; however, the Adabas nucleus continues to run. The disabled user exit is not recalled until it is reset. Once the exit error has been corrected, the exit can be reactivated using the XACTIVATE operand.

The XCRITICAL (XC) operand changes the status of an exit from "noncritical" back to the default "critical" for the functioning of the Adabas nucleus. Any abnormal termination or program check in a critical exit causes the Adabas nucleus to terminate.

```
SMGT, XLOAD= {
  exit-code
  (exit-code, module-name)
}
```

The XLOAD (XL) operand loads a new version of an exit module.

The "module-name" parameter is required only for new exits. Because the default is the previous module name, the parameter is optional for exits that have previously been used. The parameter is also optional for any exit defined with a number of zero; for example, the ADACSH exit which is always named ADACSHUX.

SMGT Display Output Samples

This section provides sample output for the DISPLAY operand of the SMGT command.

SMGT,DISPLAY = {ALL | EXITS | MSGBUF | PINS | SUMMARY | LAST }

The DISPLAY (D) operand writes status and history information about the Adabas error handling and message buffering facility to the job log and to DDPRINT:

ALL	(the default) displays all reports
EXITS	displays current user exit, hyperexit, and other exit status
MSGBUF	displays message buffering status
PINS	displays PIN routine status
SUMMARY	displays only the summary information from ALL
LAST	displays details of the most recent recovery action

For more information about the messages displayed using this command, see the Adabas Messages and Codes documentation.

DISPLAY=ALL

The ALL parameter displays all of the information shown for the other parameters in the following paragraphs.

DISPLAY=EXITS

```
ADANA1 00127 SMGT DISPLAY ACTIVE
ADANA2 00127 SMGT ACTIVE
ADANAX 00127 EXIT: UX04 MODNAME: EXIT1 STATUS: ACTIVE
ADANAX 00127 EXIT: HX22 MODNAME: NULLEXIT STATUS: ACTIVE
ADANO2 00127 SMGT COMMAND PROCESSED
```

DISPLAY=MSGBUF

```
ADANA1 00127 SMGT DISPLAY ACTIVE
ADANA2 00127 SMGT ACTIVE
ADANAE 00127 MESSAGE BUFFERING IS ACTIVE
ADANO2 00127 SMGT COMMAND PROCESSED
```

DISPLAY=PINS

```
ADANA1 00127 SMGT DISPLAY ACTIVE
ADANA2 00127 SMGT ACTIVE
ADANAG 00127 PIN 0001 USES: 0000 CONDITION: 000C4000
                THIS PIN VALID FOR ALL LOCATIONS
ADANAG 00127 PIN 0002 USES: 0000 CONDITION: 000C1000
                LOCATION: 00081C6C 0008259B (EBL2 )
ADANAA 00127 002 CONDITION PIN ROUTINES RECOVERED 000 ERRORS
ADANAB 00127 000 LOCATION PIN ROUTINES RECOVERED 000 ERRORS
ADANAG 00127 PIN 0003 USES: 0000 CONDITION: RSP: 017
```

```
                THIS PIN VALID FOR ALL LOCATIONS
ADANAC 00127 001 RESPONSE PIN ROUTINES RECOVERED 000 ERRORS
ADANAD 00127 003 TOTAL PIN ROUTINES RECOVERED 000 ERRORS
ADANA8 00127 000 EXECUTIONS OF ABNORMAL TERMINATION HANDLER
ADANA8 00127 000 EXECUTIONS OF PROGRAM CHECK HANDLER
ADANA8 00127 000 EXECUTIONS OF RESPONSE CODE HANDLER
ADANO2 00127 SMGT COMMAND PROCESSED
```

DISPLAY=SUMMARY

```
ADANA1 00127 SMGT DISPLAY ACTIVE
ADANA2 00127 SMGT ACTIVE
ADANA3 00127 ABNORMAL TERMINATION HANDLER ACTIVE
ADANA4 00127 PROGRAM CHECK HANDLER ACTIVE
ADANAA 00127 002 CONDITION PIN ROUTINES RECOVERED 000 ERRORS
ADANAB 00127 000 LOCATION PIN ROUTINES RECOVERED 000 ERRORS
ADANAC 00127 001 RESPONSE PIN ROUTINES RECOVERED 000 ERRORS
ADANAD 00127 003 TOTAL PIN ROUTINES RECOVERED 000 ERRORS
ADANA8 00127 000 EXECUTIONS OF ABNORMAL TERMINATION HANDLER
ADANA8 00127 000 EXECUTIONS OF PROGRAM CHECK HANDLER
ADANA8 00127 000 EXECUTIONS OF RESPONSE CODE HANDLER
ADANO2 00127 SMGT COMMAND PROCESSED
```

DISPLAY=LAST

No error:

```
ADANA1 00127 SMGT DISPLAY ACTIVE
ADANA2 00127 SMGT ACTIVE
ADANA5 00127 NO ERROR CONDITIONS HANDLED
ADANO2 00127 SMGT COMMAND PROCESSED
```

Error Encountered:

```
ADANA1 00127 SMGT DISPLAY ACTIVE
ADANA2 00127 SMGT ACTIVE
ADANA6 00127 LAST ERROR OCCURRED AT: 1998-07-07 14:36:18
ADANA7 00127 CONDITION: RSP: 017 LOCATION: *N/A*
ADANO2 00127 SMGT COMMAND PROCESSED
```

Adabas Session Execution

An Adabas session involves the execution of the Adabas nucleus which controls access/update to a single database. This chapter describes the job control statements needed when executing an Adabas session under each supported operating system. For examples of the Adabas utility jobs, see the Adabas Utilities documentation.

This chapter covers the following topics:

- Program Synchronization Using Operating System Services
- BS2000 Session
- OS/390 or z/OS Session
- VM/ESA or z/VM Session
- VSE/ESA Session
- Notes
- Single-User Mode
- User Profile

Program Synchronization Using Operating System Services

Adabas version 7 uses operating system services to synchronize the start and end of nucleus and utility executions. Only one program can modify the data integrity block (DIB) at a time.

The operating system services used are as follows:

Operating System	Service
OS/390 and z/OS	systems-wide ENQ/DEQ macros (SCOPE=SYSTEMS) with major name (QNAME) 'ADABAS'
BS2000	system-wide ENQAR/DEQAR macros (SCOPE=GLOBAL)
VSE/ESA	system-wide LOCK/UNLOCK macros
VM/ESA and z/VM	system-wide ENQ/DEQ macros (SCOPE=SYSTEMS) with major name (QNAME) 'ADABAS'

This feature reliably and efficiently guarantees proper synchronization of DIB updates within a single operating-system image.

If your database resides on disks that are shared among multiple images of the operating system and you run nucleus or utility jobs against the same database on more than one of the system images, you need to ensure that

- the system images are installed in such a way that synchronization is effective on all systems where nucleus and utility jobs execute; or
- nucleus and utility jobs do not execute concurrently on different system images.

Consult your system programmer for the needed information.



Warning:

If different nucleus or utility jobs updating the same file are allowed to start or terminate on different system images at the same time without proper synchronization, a DIB update may be lost. If this happens, a lock in the DIB may be violated, thereby opening the file to the possibility of destruction due to concurrent unsynchronized updates by utilities.

BS2000 Session

The following table contains all datasets which are used when executing an Adabas session under BS2000.

Dataset	Link Name	Logical Unit	Storage Medium	Additional Information
ADARUN parameters	DDCARD	SYSDTA	disk	note 1
ADARUN / Adabas messages	DDPRINT	SYSOUT	disk	note 2
Associator	DDASSORn		disk	note 3
Data Storage	DDDATARn		disk	note 3
Work	DDWORKR1		disk	note 4
Recovery Aid log	DDRLOGR1		disk	note 5
Protection log multiple log 1 multiple log 2	DDSIBA DDPLOGR1 DDPLOGR2		tape disk disk disk	note 6 note 7 note 7
Command log multiple log 1 multiple log 2	DDLOG DDCLOGR1 DDCLOGR2		tape disk disk disk	note 8 note 9 note 9
ECS encoding objects	DDECSOJ		tape/disk	note 10

Example of Adabas Session Job Control (BS2000)

This job includes multiple protection logging, multiple command logging, Recovery Aid logging, and universal encoding support (UES):

In SDF Format:

```

/.ADANUC LOGON
/MODIFY-TEST-OPTIONS DUMP=YES
/REMARK *
/REMARK * ADABAS NUCLEUS
/REMARK *
/ASS-SYSLST L.NUC
/ASS-SYSDTA *SYSCMD
/SET-FILE-LINK DDLIB,ADAvrs.MOD
/SET-FILE-LINK DDASSOR1,ADAYyyyy.ASSO,SHARE-UPD=YES
/SET-FILE-LINK DDDATAR1,ADAYyyyy.DATA,SHARE-UPD=YES
/SET-FILE-LINK DDWORKR1,ADAYyyyy.WORK
/SET-FILE-LINK DDPLOGR1,ADAYyyyy.PLOGR1,SHARE-UPD=YES
/SET-FILE-LINK DDPLOGR2,ADAYyyyy.PLOGR2,SHARE-UPD=YES
/SET-FILE-LINK DDCLOGR1,ADAYyyyy.CLOGR1,SHARE-UPD=YES
/SET-FILE-LINK DDCLOGR2,ADAYyyyy.CLOGR2,SHARE-UPD=YES
/SET-FILE-LINK DDRLOGR1,ADAYyyyy.RLOGR1,SHARE-UPD=YES
/START-PROGRAM *M(ADA.MOD,ADARUN),PR-MO=ANY
ADARUN PROG=ADANUC,DB=yyyyyy
ADARUN LBP=600000
ADARUN LWP=320000
ADARUN LS=80000
ADARUN LP=400
ADARUN NAB=24
ADARUN NC=50
ADARUN NH=2000
ADARUN NU=100
ADARUN TNAE=180,TNAA=180,TNAX=600,TT=90
ADARUN NPLOG=2,PLOGSIZE=1800,PLOGDEV=dddd
ADARUN NCLOG=2,CLOGSIZE=1800,CLOGDEV=dddd
/REMARK
/LOGOFF SYS-OUTPUT=DEL

```

In ISP Format:

```

/.ADANUC LOGON
/OPTION MSG=FB,DUMP=YES
/REMARK *
/REMARK * ADABAS NUCLEUS
/REMARK *
/SYSFILE SYSLST=L.NUC

/FILE ADAvrs.MOD,LINK=DDLIB
/FILE EXAMPLE.ADAYyyyy.ASSOR1,LINK=DDASSOR1,SHARUPD=YES
/FILE EXAMPLE.ADAYyyyy.DATAR1,LINK=DDDATAR1,SHARUPD=YES
/FILE EXAMPLE.ADAYyyyy.WORKR1,LINK=DDWORKR1
/FILE EXAMPLE.ADAYyyyy.PLOGR1,LINK=DDPLOGR1,SHARUPD=YES
/FILE EXAMPLE.ADAYyyyy.PLOGR2,LINK=DDPLOGR2,SHARUPD=YES
/FILE EXAMPLE.ADAYyyyy.CLOGR1,LINK=DDCLOGR1,SHARUPD=YES
/FILE EXAMPLE.ADAYyyyy.CLOGR2,LINK=DDCLOGR2,SHARUPD=YES
/FILE EXAMPLE.ADAYyyyy.RLOGR1,LINK=DDRLOGR1,SHARUPD=YES
/EXEC (ADARUN,ADAvrs.MOD)
ADARUN PROG=ADANUC,DB=yyyyyy
ADARUN LBP=600000
ADARUN LWP=320000
ADARUN LS=80000
ADARUN LP=400
ADARUN NAB=24
ADARUN NC=50
ADARUN NH=2000
ADARUN NU=100

```

```
ADARUN TNAE=180 ,TNA=180 ,TNAX=600 ,TT=90
ADARUN NPLOG=2 ,PLOGSIZE=1800 ,PLOGDEV=dddd
ADARUN NCLOG=2 ,CLOGSIZE=1800 ,CLOGDEV=dddd
/REMARK
/LOGOFF NOSPOOL
```

JCL Required for UES Support (BS2000)

If you are using universal encoding support (UES), the following additional JCL is required for BS2000 environments:

In SDF Format:

```
/SET-FILE-LINK DDEC SOJ,ADAvrs.ALLECSO
```

In ISP Format:

```
/FILE ADAVrs.ALLECSO, LINK=DDEC SOJ
```

OS/390 or z/OS Session

The following datasets are required when executing an Adabas session under OS/390 or z/OS.

Dataset	Dataset Name	Storage Medium	Additional Information
ADARUN parameters	DDCARD	card image	note 1
ADARUN / Adabas messages	DDPRINT	printer	note 2
Associator	DDASSORn	disk	note 3
Data Storage	DDDATARn	disk	note 3
Work	DDWORKR1	disk	note 4
Recovery Aid log	DDRLOGR1	disk	note 5
Protection log	DDSIBA	tape/disk	note 6
multiple log 1	DDPLOGR1	disk	note 7
multiple log 2	DDPLOGR2	disk	note 7
Command log	DDLOG	tape/disk	note 8
multiple log 1	DDCLOGR1	disk	note 9
multiple log 2	DDCLOGR2	disk	note 9
ECS encoding objects	DDEC SOJ	tape/disk	note 10
Abnormal termination	MPMDUMP	printer	note 11
SMGT dump and snap dump	ADASNAP	printer	note 12

Example of Adabas Session Job Control (OS/390 and z/OS)

This job includes multiple protection logging, multiple command logging, and Recovery Aid logging:

```
//NUC099      EXEC PGM=ADARUN
//STEPLIB    DD DISP=SHR,DSN=ADABAS.Vvrs.ADAvrs.MVSLOAD
//DDASSOR1   DD DISP=SHR,DSN=EXAMPLE.ADAyyyyy.ASSOR1
//DDDATAR1   DD DISP=SHR,DSN=EXAMPLE.ADAyyyyy.DATAR1
//DDWORKR1   DD DISP=OLD,DSN=EXAMPLE.ADAyyyyy.WORKR1
//DDPLOGR1   DD DISP=SHR,DSN=EXAMPLE.ADAyyyyy.PLOGR1
//DDPLOGR2   DD DISP=SHR,DSN=EXAMPLE.ADAyyyyy.PLOGR2
//DDCLOGR1   DD DISP=SHR,DSN=EXAMPLE.ADAyyyyy.CLOGR1
//DDCLOGR2   DD DISP=SHR,DSN=EXAMPLE.ADAyyyyy.CLOGR2
//DDRLOGR1   DD DISP=SHR,DSN=EXAMPLE.ADAyyyyy.RLOGR1
//DDPRINT    DD SYSOUT=X
//SYSUDUMP   DD SYSOUT=X
//MPMDUMP    DD SYSOUT=X
//ADASNAP    DD SYSOUT=X
//DDCARD     DD *
ADARUN PROG=ADANUC,DB=yyyyy
ADARUN LBP=600000
ADARUN LWP=320000
ADARUN LS=80000
ADARUN LP=400
ADARUN NAB=24
ADARUN NC=1000
ADARUN NH=2000
ADARUN NU=100
ADARUN TNAE=180,TNAA=180,TNAX=600,TT=90
ADARUN NPLOG=2,PLOGSIZE=1800,PLOGDEV=dddd
ADARUN NCLOG=2,CLOGSIZE=1800,CLOGDEV=dddd
//
```

where

dddd is a valid device type.

nn is the load library level. If the library with a higher level number is not a full replacement for the lower level load library(s), the library with the higher level must precede those with lower numbers in the steplib concatenation.

vrs is the version, revision, and system maintenance level of the product.

yyyyy is the physical database ID.

JCL Required for UES Support (OS/390 and z/OS)

If you are using universal encoding support (UES), you must

- include the following additional libraries for internal products in the steplib:

```
//STEPLIB DD DISP=SHR,DSN=ADABAS.Vvrs.BTEvrs.MVSLDnn
//          DD DISP=SHR,DSN=ADABAS.Vvrs.APSvrs.MVSLDnn
```

where "nn" is the load library level. If the library with a higher level number is not a full replacement for the lower level load library(s), the library with the higher level must precede those with lower numbers in the steplib concatenation.

Note:

If you are using an Adabas load library prior to version 7.2.2, it contains internal product libraries with an earlier version number and must be ordered below the current internal product libraries in the steplib concatenation.

- add the following additional JCL related to internal product libraries:

```
//DDECSOJ DD DISP=SHR,DSN=ADABAS.Vvrs.BTEvrs.MVSEC SO
//SYSPARM DD *
SYSTEM_ID=ADAAPS
ABEND_RECOVERY=NO
TRHEAD_ABEND_RECOVERY=NO
```

JCL Required for UES and TCP/IP Support (OS/390 and z/OS)

If you are connecting your UES-enabled database directly through a TCP/IP link, you must also

- include the ADATCP library in the steplib:

```
//STEPLIB DD ....
//          DD DISP=SHR,DSN=ADABAS.Vvrs.WATvrs.MVSLOAD
```

- identify the TCP/IP stack you intend to use with the CDI_DRIVER parameter of the SYSPARM statement:

```
//DDECSOJ DD DISP=SHR,DSN=ADABAS.Vvrs.BTEvrs.MVSEC SO
//SYSPARM DD *
SYSTEM_ID=ADAAPS
ABEND_RECOVERY=NO
TRHEAD_ABEND_RECOVERY=NO
* User must choose one of the following depending on the TCP/IP stack used:
*CDI_DRIVER=('tcpip,PAALSOCK,SUBSYS=ACSS')      <--Interlink TCP/IP stack
*CDI_DRIVER=('tcpip,PAALSOCK,ADDRSPCE=STACKNAME') <--IBM TCP/IP stack for HPS
*CDI_DRIVER=('tcpip,PAALSOCK,ADDRSPCE=STACKNAME') <--IBM TCP/IP stack for OE
```

VM/ESA or z/VM Session

The following table contains all datasets which are used when executing an Adabas session under VM/ESA.

Dataset	Dataset Name	Storage Medium	Additional Information
ADARUN parameters	DDCARD	card image	note 1
ADARUN/Adabas messages	DDPRINT	printer	note 2
Associator	DDASSORn	disk	note 3
Data Storage	DDDATARn	disk	note 3
Work	DDWORKR1	disk	note 4
Recovery Aid log	DDRLOGR1	disk	note 5
Protection log	DDSIBA	tape/disk	note 6
multiple log 1	DDPLOGR1	disk	note 7
multiple log 2	DDPLOGR2	disk	note 7
Command log	DDLOG	tape/disk	note 8
multiple log 1	DDCLOGR1	disk	note 9
multiple log 2	DDCLOGR2	disk	note 9
ECS encoding objects	DDECSOJ	tape/disk	note 10
Abnormal termination	-	printer	note 11

Example of Adabas Session Job Structure (VM/ESA or z/VM)

This job includes multiple protection logging, multiple command logging, Recovery Aid logging, and universal encoding support (UES):

```

DATADEF DDASSOR1,DSN=EXAMPLE.ADAyyyyy.ASSOR1,VOL=ASSOV1
DATADEF DDDATAR1,DSN=EXAMPLE.ADAyyyyy.DATAR1,VOL=DATAV1
DATADEF DDWORKR1,DSN=EXAMPLE.ADAyyyyy.WORKR1,VOL=WORKV1
DATADEF DDPLOGR1,DSN=EXAMPLE.ADAyyyyy.PLOGR1,VOL=PLOGV1
DATADEF DDPLOGR2,DSN=EXAMPLE.ADAyyyyy.PLOGR2,VOL=PLOGV2
DATADEF DDCLOGR1,DSN=EXAMPLE.ADAyyyyy.CLOGR1,VOL=CLOGV1
DATADEF DDCLOGR2,DSN=EXAMPLE.ADAyyyyy.CLOGR2,VOL=CLOGV2
DATADEF DDRLOGR1,DSN=EXAMPLE.ADAyyyyy.RLOGR1,VOL=RLOGV1
DATADEF DDPRINT,DSN=ADANUC.DDPRINT,MODE=A
DATADEF DUMP,DUMMY
DATADEF DDCARD,DSN=RUNNUC.CONTROL,MODE=A
ADARUN

```

CONTENTS OF RUNNUC.CONTROL.A1 :

```

ADARUN PROG=ADANUC,DEVICE=dddd,DB=yyyyy
ADARUN LBP=60000
ADARUN LWP=32000
ADARUN LS=80000
ADARUN LP=400
ADARUN NAB=24
ADARUN NC=1000
ADARUN NH=2000
ADARUN NU=100
ADARUN TNAE=180,TNAA=180,TNAX=600,TT=90
ADARUN NPLOG=2,PLOGSIZE=1800,PLOGDEV=dddd
ADARUN NCLOG=2,CLOGSIZE=1800,CLOGDEV=dddd

```

Note:

See the Adabas Installation documentation for more information about Adabas operation under VM/ESA or z/VM.

JCL Required for UES Support (VM/ESA or z/VM)

If you are using universal encoding support (UES), the following additional JCL is required for VM/ESA environments:

```
DATADEF DDECSOJ,DSN=ADAvrs.ALLECSO,VOL=ECSOBJ
```

VSE/ESA Session

The following table contains all datasets used when executing an Adabas session under VSE/ESA. "SYSnnn" means that any programmer logical unit may be used.

Dataset	File Name	Logical Unit	Storage Medium	Additional Information
ADARUN parameters	none CARD CARD	SYSRDR SYS000 SYSnnn	reader tape disk	note 1
ADARUN/Adabas messages	none	SYSLST	printer	note 2
Associator	ASSORn	SYSnnn	disk	note 3
Data Storage	DATARn	SYSnnn	disk	note 3
Work	WORKR1	SYSnnn	disk	note 4
Recovery Aid log	RLOGR1	SYSnnn	disk	note 5
Protection log multiple log 1 multiple log 2	SIBA PLOGR1 PLOGR2	SYS014 SYSnnn SYSnnn SYSnnn	tape disk disk disk	note 6 note 7 note 7
Command log multiple log 1 multiple log 2	LOG CLOGR1 CLOGR2	SYS012 SYSnnn SYSnnn SYSnnn	tape disk disk disk	note 8 note 9 note 9
ECS encoding objects	DDECSOJ	SYS020 *	tape disk	note 10

Example of Adabas Session Job Control (VSE/ESA)

This job includes multiple protection logging, multiple command logging, and Recovery Aid logging:

```
// ASSGN SYS031,dddd,VOL=ADA001,SHR
// ASSGN SYS032,dddd,VOL=ADA002,SHR
// ASSGN SYS033,dddd,DISK,VOL=ADA003,SHR
// ASSGN SYS034,dddd,VOL=ADA004,SHR
// DLBL
```

```

ASSOR1,'EXAMPLE.ADAyyyyy.ASSOR1',2099/365,DA
// EXTENT SYS031,ADA001,,,15,1500
// DLBL DATAR1,'EXAMPLE.ADAyyyyy.DATAR1',2099/365,DA
// EXTENT SYS032,ADA002,,,15,3000
// DLBL WORKR1,'EXAMPLE.ADAyyyyy.WORKR1',2099/365,DA
// EXTENT SYS033,ADA003,,,15,600
// DLBL PLOGR1,'EXAMPLE.ADAyyyyy.PLOGR1',2099/365,DA
// EXTENT SYS034,ADA004,,,15,600
// DLBL PLOGR2,'EXAMPLE.ADAyyyyy.PLOGR2',2099/365,DA
// EXTENT SYS034,ADA004,,,615,600
// DLBL CLOGR1,'EXAMPLE.ADAyyyyy.CLOGR1',2099/365,DA
// EXTENT SYS034,ADA004,,,1215,600
// DLBL CLOGR2,'EXAMPLE.ADAyyyyy.CLOGR2',2099/365,DA
// EXTENT SYS034,ADA004,,,1815,600
// DLBL RLOGR1,'EXAMPLE.ADAyyyyy.RLOGR1',2099/365,DA
// EXTENT SYS034,ADA004,,,1300,600
// DLBL ADAvCL,'ADABAS.Vvrs.LOADLIB',2099/365
// EXTENT ,ADADSK
// EXEC ADARUN,SIZE=ADARUN
ADARUN PROG=ADANUC,SVC=xxx,DEVICE=dddd,DB=yyyyy
ADARUN LBP=600000
ADARUN LWP=320000
ADARUN LS=80000
ADARUN LP=400
ADARUN NAB=24
ADARUN NC=1000
ADARUN NH=2000
ADARUN NU=100
ADARUN TNAE=180,TNAA=180,TNAX=600,TT=90
ADARUN NPLOG=2,PLOGSIZE=1800,PLOGDEV=dddd
ADARUN NCLOG=2,CLOGSIZE=1800,CLOGDEV=dddd
/*
/*

```

JCL Required for UES Support (VSE/ESA)

The following additional JCL is required for universal encoding support (UES):

```

// ASSGN SYS020,disk,VOL=volume,SHR
// DLBL DDECSOJ,'ADABAS.Vvrs.ECSLIB'
// EXTENT SYS020

```

Notes

1.

This dataset is used to provide the Adabas session parameters.

2.

This dataset is used to print messages produced by the control module ADARUN and/or the Adabas nucleus.

3.

The Adabas Associator and Data Storage. These datasets are mandatory.

"n" represents the number of the Associator and Data Storage dataset, respectively. If only one dataset exists for each, "n" must be "1". If more than one dataset exists for Associator and/or Data Storage, a separate statement is required for each. For example, if the Associator consists of two datasets, the statements DD/ASSOR1 and DD/ASSOR2 are required.

4.

The Adabas Work dataset. This dataset is mandatory.

Software AG recommends running the nucleus with DISP=OLD (under OS/390 or z/OS; "share" not specified for BS2000 and VSE/ESA) for the WORKR1 dataset as a way of preventing two nuclei from writing to the same WORK dataset and corrupting the database. This could otherwise happen if the ADARUN parameters FORCE and IGNDIB are improperly used.

5.

If the Adabas Recovery Aid is being used, this logging dataset is required.

6.

The data protection log dataset. This dataset is required if the database will be updated during the session and logging of protection information is desired. This dataset is not applicable if multiple protection logging is used.

The data protection log may be assigned to tape or disk. A new dataset must be used for each Adabas session (DISP=MOD may not be used). See *Adabas Restart and Recovery* for additional information.

7.

Multiple (two to eight) data protection log datasets. These datasets are required only if multiple data protection logging is to be in effect for the session.

Multiple data protection logging is activated by the ADARUN NPLOG and PLOGSIZE parameters. The device type of the multiple protection logs is specified with the ADARUN PLOGDEV parameter.

Whenever one of multiple protection log datasets is full, Adabas switches automatically to another dataset and notifies the operator through a console message that the log which is full should be copied using the PLCOPY function of the ADARES utility. This copy procedure may also be implemented using the user exit 12 facility as described in the User Exits documentation.

If no command logging is to be performed, this dataset may be omitted.

8.

The command log dataset. This dataset is required if command logging is to be performed during the session. Command logging is activated by the ADARUN LOGGING parameter.

9.

Multiple (two to eight) command log datasets. These datasets are required only if multiple command logging is to be in effect for the session.

Multiple command logging is activated by the ADARUN NCLOG and CLOGSIZE parameters. The device type of the multiple command log datasets is specified with the ADARUN CLOGDEV parameter.

Whenever one of multiple command log datasets is full, Adabas switches automatically to another dataset and notifies the operator through a console message that the log which is full should be copied using the CLCOPY function of the ADARES utility. This copy procedure may also be implemented using the user exit 12 facility as described in the Adabas User Exits documentation.

10.

The Entire Conversion Services (ECS) objects dataset is required for universal encoding support (UES).

11.

This dataset is used to take an Adabas dump including SVC, ID-TABLE and allocated CSA in the event that an abnormal termination occurs.

The line count in the JCL must be set appropriately; otherwise, the dump cannot be printed in its entirety.

12.

This dataset is used under OS/390 or z/OS to take an Adabas dump (SMGT,DUMP) or snap dump (SMGT,SNAP) when using the error handling and message buffering facility.

Single-User Mode

Although the normal mode of operation is multiuser mode, it is also possible to execute Adabas together with a user program or Adabas utility in the same region.

For single-user mode, you must include the Adabas nucleus job control that you use along with the job control for the utility or user program.

The Adabas prefetch option cannot be used in single-user mode; however, single-user mode *must* be used when running a read-only nucleus and an update nucleus simultaneously.

User Profile

Some information within an Adabas database is user-related and must be retained from session to session. One such kind of information is ET data records; another is the priority value assigned to a user.

A set of user-related information can be stored in a profile table. The values stored in this table are read at OPEN time and assigned to the user. The direct call user must OPEN the Adabas session with the proper call; that is, as an ID user with an ETID in the additions 1 field of the Adabas control block. For Natural users, the profile table is identified by the Natural ETID.

The associated fields are user-related timeout and threshold values, and the OWNERID for multiclient fields. One record per user is stored. The profile table is maintained using Adabas Online System.

Profile Table Values

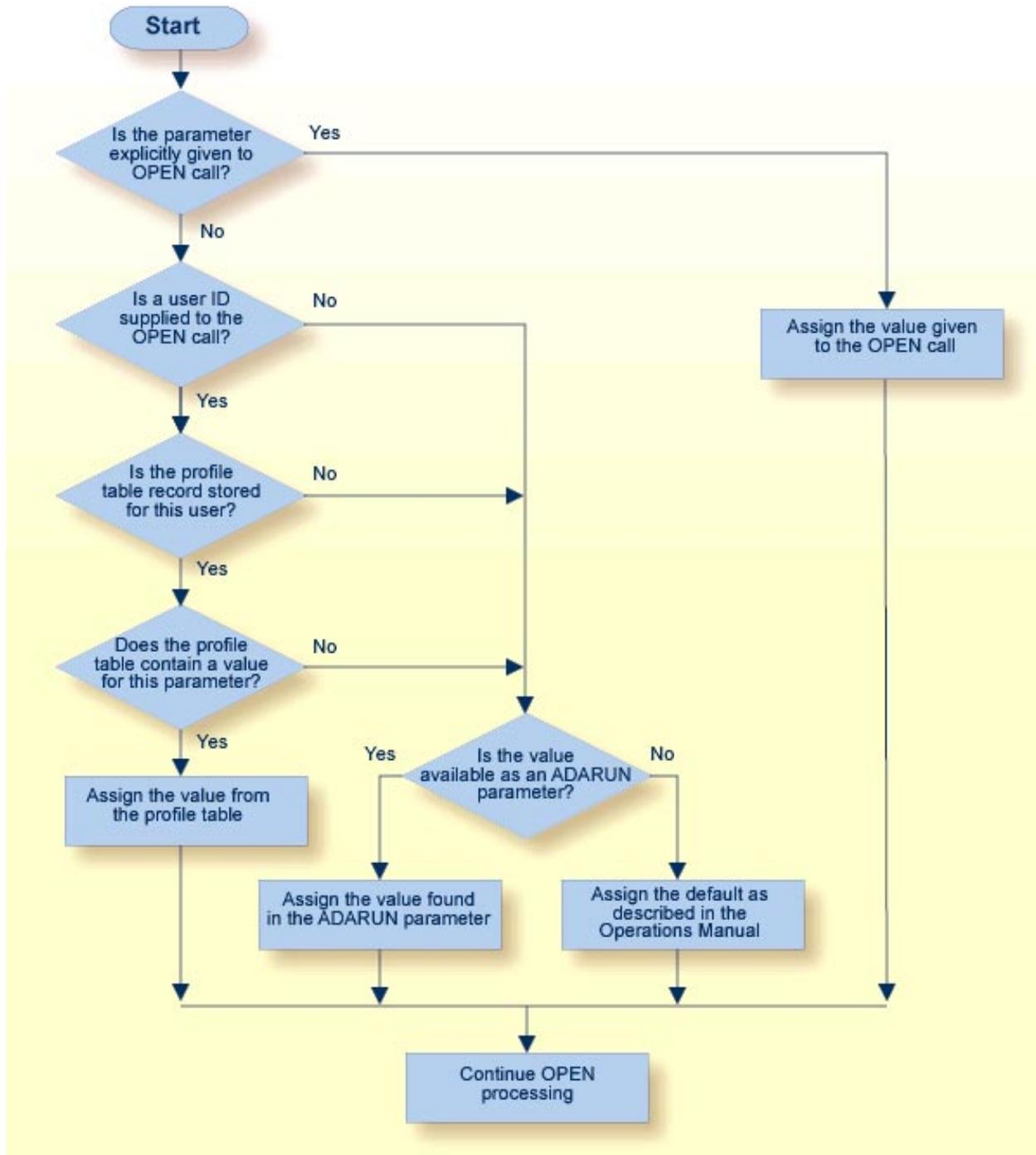
The user-related values shown below are currently stored in the profile table.

Value	Description
PRIORITY	User's priority (0-255)
TNAA*	Access user non-activity time
TNAE*	ET user non-activity time
TNAX*	EXU/EXF user non-activity time
TT*	Transaction time threshold
TLSCMD*	Sx command threshold
NSISN*	Maximum number of ISNs per TBI element
NSISNHQ*	Maximum number of records held by user
NQCID*	Maximum number of active command IDs per user
OWNERID	Owner ID for multiclient file access

* The decision sequence for determining the values for a user at the time of an open call is shown in *Managing the User Profile*.

Managing the User Profile

Adabas Online System (AOS) must be used to maintain the profile table. See the Adabas Online System documentation for detailed information about managing the profile table.



Profile Table Decision Flow

Adabas Utility Execution

The Adabas Utilities documentation contains detailed information about the function and execution of each Adabas utility.

You can use Adabas Online System to execute various utility functions online.

This chapter covers the following topics:

- Condition and Termination Codes
- File and Database Usage
- Disk Accesses by Utility and Function
- Possible Resource Conflicts
- Repetitive Utility Execution Under VM/ESA or z/VM

Condition and Termination Codes

Refer to the Adabas Messages and Codes documentation for detailed descriptions of the condition and termination code meanings for each utility.

The following condition codes can be set by an Adabas utility:

Code	The utility operation ...
0	was successful.
4	was successful but encountered a warning condition.
8	encountered an error condition but then continued.
16	was successful in its main function but then encountered an error condition.
20	could not be completed. The operation encountered an error condition but did not terminate abnormally because NOUSERABEND was specified.

For errors that occur during the execution of a utility, the following termination codes are returned:

Code	The utility terminated . . .
34	with a dump.
35	without a dump.
36	with a dump. Termination was caused by an I/O error while writing to DDDRUCK or DDPRINT. Check the JCL.

File and Database Usage

The table in this section shows the type of file/database usage required for each utility, the database status required, whether the Adabas nucleus must be active while an Adabas utility is running, and whether the function creates an entry in the data integrity block (DIB).

Any utility that requires a nucleus can be used in single-user mode, provided that the nucleus JCL statements are specified.

The following codes are used in the table:

Code	Meaning
n/a	Does not apply.
	File/Database Usage
ACC	Access-only usage: other users may access and update the file/database.
EXF	Exclusive file control: other users may not access or update the file.
EXU	Exclusive file update: other users may not update the file.
UTI	Utility update control: other users may not access or update the file.
UTS	Utility SAVE control: other users may access and update the file.
*	Exclusive database control: no other database activity is permitted.
**	Exclusive database control if the checkpoint or security file is involved; otherwise, read-only access is allowed during the read phase of utility operation.
***	Exclusive database control if the CONTINUE parameter is specified.
UPD	Update/ET user status does not apply to utilities. Utilities cannot be timed out.
	Database Status
NP	Other activity against the database is not permitted.
P	Activity against other database files is permitted.
	Adabas Nucleus Status
A	Adabas nucleus must be active.
C	Adabas nucleus may be active.
N	Adabas nucleus must not be active.

Utility	Function	File/DB Usage	Database Activity	Adabas Nucleus	DIB Entry Created
ADAACK	All	EXU	P	C	NO
ADACDC	All	n/a	P	C	NO

Utility	Function	File/DB Usage	Database Activity	Adabas Nucleus	DIB Entry Created
ADACMP	Compress FDT=...	ACC	P	A	NO
	Compress data	n/a	P	C	NO
	Decompress data	n/a		C	NO
	Decompress INFILE=...	EXU(EXF)		A	NO
ADACNV	All	*	NP	N	YES
	With TEST parameter	ACC	P	C	NO
ADADBS	All, except:	ACC/UTI	P (1)	A	NO
	RECOVER	UTI	(2)	A	NO
	RESETDIB	n/a	P	C	NO
	OPERCOR DDIB	n/a	P	C	NO
ADADCK	All	EXU	P	C	NO
ADADEF	All	*	NP	N	NO
ADAFRM	Formatting other than TEMP/SORT dataset	n/a	NP	N	NO
	Formatting TEMP/SORT dataset	n/a	P	C	NO
	Formatting single blocks	n/a	P	C	NO
ADAICK	ICHECK	EXU	P	C	NO
ADAINV	All	EXU/UTI	P	A	YES
ADALOD	All	UTI	P	C	YES
ADAMER	ADAM estimation	n/a	P	C	NO
ADAORD	REORDER A/D/DB	*	NP	A	YES
	REORDER FILE (A)	EXU/UTI	P	A	YES
	REORDER FILE (D)	**	P	A	YES
	REORDER FILE	EXU/UTI	P	A	YES
	RESTRUCTURE DB	**	NP	A	YES
	RESTRUCTURE FILE	EXU/UTI	P	A	YES
	STORE	**	P	A	YES
		* EXU ** UTI			
ADAPLP	All	n/a	P	C	NO
ADAPRI	Maint print	n/a	P	C	NO
ADARAI	DISABLE, PREPARE, REMOVE CHKDB, LIST	n/a	P	N	NO
	RECOVER	n/a	P	C	NO

Utility	Function	File/DB Usage	Database Activity	Adabas Nucleus	DIB Entry Created
ADAREP	Database reporting CPLIST/CPEXLIST	n/a	P	C	NO
		ACC	P	A	NO
ADARES	Regenerate file,	UTI ***	P	A	NO
	Backout file	*	NP	A	NO
	Regenerate,	n/a	P	A/C	NO
	Backout	n/a	P	C	NO
	Copy	UTI	P	A	NO
	Copy dual/multi log Repair				
ADASAV	Save file	UTS	P	C	YES
	Save database	UTS	P (3)	C	YES
	Restore file	UTI	P	C	YES
	Restore database	*	NP	N	YES
	Restore protection log	*	NP	N	YES
ADASEL	Select protection log data	n/a	P	C	NO
ADAULD	Unload file Unload file from save tape	EXU(EXF) n/a	P P	A C	NO NO
ADAVAl	Validate	EXU	P	A	NO
ADAZAP	All	*	NP	N	NO

Notes:

1. When completed, ADADBS ADD/INCREASE end the nucleus session to permit reallocation and formatting.
2. ADADBS RECOVER cannot run when other utilities are running.
3. When running ADASAV SAVE with an active nucleus ("online"), no other utilities can be run which update the database.

Disk Accesses by Utility and Function

The following table lists the datasets that each utility function accesses on disk:

Utility	Function	Disk Accessed Directly
ADAACK	all	ASSO, DATA
ADACDC	all	ASSO
ADACMP	all	n/a
ADACNV	all	ASSO, DATA, WORK, dual PLOG if specified
ADADBS	OPERCOR DDIB, RESETDIB others	ASSO, if nucleus is down n/a
ADADCK	all	ASSO, DATA

Utility	Function	Disk Accessed Directly
ADADEF	DEFINE NEWWORK	ASSO, DATA, WORK ASSO, WORK
ADAFRM	all	the dataset to be formatted
ADAICK	DATAPRINT, DSCHECK others	ASSO, DATA ASSO
ADAINV	all	ASSO
ADALOD	all	ASSO, DATA, WORK
ADAMER	all	n/a
ADAORD	REORDB, REORFILE, REORDATA, REORFDATA, STORE REORASSO, REORFASSO, RESTRUCTUREDB, RESTRUCTUREF	ASSO, DATA ASSO
ADAPLP	WORKPRI PLOGPRI others	WORK Dual PLOG n/a
ADAPRI	all	ASSO + dataset to be printed
ADARAI	PREPARE, REMOVE all others	ASSO n/a
ADAREP	all report from SAVETAPE*	ASSO; DATA is not required but is recommended because ADAREP tries to print the volume numbers. n/a
ADARES	CLCOPY, PLCOPY all others	Dual CLOG, PLOG; ASSO is recommended but not required ASSO
ADASAV	SAVE DB, SAVE FILE, RESTORE FILE	ASSO, DATA; also WORK if nucleus is down
ADASEL	all	ASSO
ADAULD	all	n/a
ADAVAl	all	n/a
ADAZAP	all ADAZAP is to DATA ADAZAP is to WORK	ASSO DATA WORK

* Adabas makes no attempt to open DATA and ASSO for an ADAREP from a SAVETAPE. For a regular ADAREP, an attempt is made to open DATA.

Possible Resource Conflicts

Note:

See *Program Synchronization Using Operating System Services* for information about how Adabas 7 uses operating system services to ensure that the start and end of nucleus and utility jobs are synchronized.

An Adabas utility which requires exclusive control of one or more Adabas files will be terminated with an error message under any of the following conditions:

1. The file has been locked using the operator command LOCKF (or Adabas Online System).

In this case, you must determine why the file was locked and whether it can now be unlocked. The UNLOCKF operator command (or Adabas Online System) may be used to unlock a locked file.

2. The file is marked as "in use" by another Adabas utility.

Adabas maintains a list of the files used by each Adabas utility in the DIB block (block 3 of the Associator). The DDIB operator command (or Adabas Online System) may be used to display this block to determine which job is using the file in question.

It is important to note that Adabas will remove entries in the DIB only when a utility terminates normally or with an error message. If a utility terminates abnormally, the files used by that utility remain "in use". The DBA may explicitly release any such files with the RESETDIB function of the ADADBS utility. For example, if the job name of the utility using the file is "JOBUT1", the following ADADBS parameters would be entered:

```
ADADBS RESETDIB JOBNAME=JOBUT1
```

If the jobname is not unique, the IDENT parameter must be used to specify the job name qualifier for the abnormally ended utility. The DDIB command (or Adabas Online System) can be used to display the identifier:

```
ADADBS RESETDIB IDENT=identifier
```

The DIB may also be reset using Adabas Online System.



Warning:

The DIB should be reset only if the utility in question actually terminated abnormally. Resetting the DIB for an active utility will cause unpredictable results.

3. The file is marked as in use by another application (user).

Adabas creates a user queue element (UQE) when the first Adabas command is received from a user program or Adabas utility. The UQE contains user information (user ID, user type, etc.) and a file list. The file list identifies each file the user is using and the type of file usage. The file-usage type is one of the following:

Type	The file is opened for . . .
ACC	access only
EXU	exclusive update
EXF	exclusive use
UPD	update (ET logic user)

See the table in the section *File and Database Usage* for information about file usage by Adabas utilities.

The file list is deleted as a result of the STOPU operator command (or using Adabas Online System), when an automatic timeout occurs, or when the user issues an Adabas CL (close) command. ET/BT users file lists are released when the user is at ET status.

Utility functions that modify the database cannot run if there is a pending autorestart, *and* the nucleus is not active.

A pending autorestart condition can only be recovered by starting the Adabas nucleus. If the nucleus cannot execute the autorestart successfully, the only valid restart method is to run

- ADASAV RESTORE (DATABASE); and then
- ADARES REGENERATE.

Repetitive Utility Execution Under VM/ESA or z/VM

To run the same utility (ADAREP, for example) more than once from an EXEC, the use of EXECOS is strongly recommended; otherwise, utility-dependent program checks can occur. The following example shows how the ADAREP utility could be run under EXECOS:

```
'EXECOS EXEC ADAREP REPCPLST 1'
'NUCXDROP ADARUN'
'EXECOS EXEC ADAREP REPCPLST 1'
```

Linking Applications to Adabas

Since most systems do not allow a standard call to Adabas, Software AG provides an application programming interface (API) to translate calls issued by an application program into a form that can be handled by Adabas.

Batch applications are supported in both single-user and multiuser mode; online operations are controlled by teleprocessing (TP) monitors. The Adabas API is available across all supported mainframe platforms; versions of the API that are specific to particular TP monitors are provided.

"Adalink" is a generic term that refers to the portion of the API that is specific to a particular TP monitor.

This chapter covers the following topics:

- How the Adabas API Works
 - Available Link Routines
 - Required Work Area
 - Required Application Reentrancy Properties
 - Adabas Control Block Options
 - Programming Conventions for Issuing Direct Calls
 - Using the Adabas API in Batch Mode
 - Support for OpenEdition OS/390 Adabas Clients
-

How the Adabas API Works

Online Operation

As an online operation, a request to Adabas is processed as follows:

1. The TP monitor invokes the application program. The application program must be loaded into the TP monitor region.
2. The application program invokes the Adabas API. *The Adabas API module must be installed in the TP monitor as an application module.*
3. The Adabas API takes the Adabas command passed to it from the application program and
 - builds the required control blocks and structures;
 - translates the Adabas parameter list provided by the application program call into a request that can be handled by the Adabas router or SVC;

- includes information that identifies the user (terminal ID, TJID etc.) to Adabas.

The TP monitor's equivalent of the LINK function is used to pass the user's Adabas control block and buffers to the API.

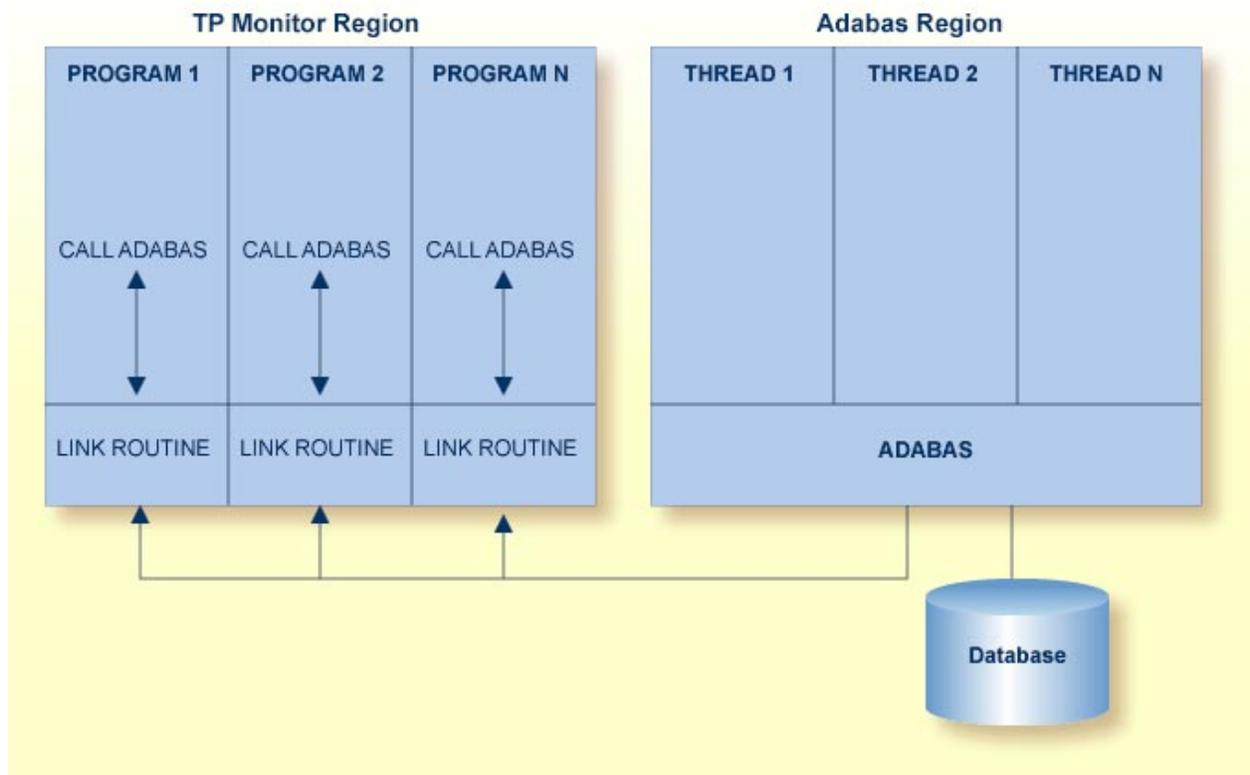
4. The Adabas API usually uses the Adabas router or SVC (supervisor call) installed on the operating system to send the formatted request to Adabas.
5. The Adabas router or SVC moves the user's control block and buffers from the TP monitor region to the Adabas region (into the Adabas nucleus).
6. The Adabas API waits for a response from the Adabas nucleus.

The TP monitor's equivalent of a WAIT is posted when the Adabas function is complete.

7. Adabas performs the function requested, then invokes the Adabas router or SVC, which returns the resulting data and response codes to the user application buffer.
8. The Adabas router or SVC then posts the Adabas API in the TP monitor region.
9. The Adabas API sends the response from the Adabas nucleus to the application program and returns control to the application program.
10. The application program returns control to the TP monitor.

The specific way each of the above functions is handled depends on the TP monitor used. In addition to these functions, each link routine can call one or more user exits at various processing points to provide additional capability and flexibility when making Adabas requests.

The following figure illustrates the basic configuration used by the majority of systems:



Adabas/TP Monitor Communication

Batch Operation

As a batch operation, a request to Adabas is processed as follows:

1. The operating system loads the batch application.
2. The batch application makes an Adabas request (CALL ADABAS ...).
3. The Adabas "stub" program ADAUSER loads and invokes ADARUN, which then loads and invokes the batch Adabas link routine ADALNK.
4. The ADALNK routine transforms the Adabas request into a format suitable for use by the Adabas nucleus.
5. The link routine invokes Adabas, usually through a call to the Adabas router or SVC installed in the operating system. It also determines a unique identification for the user.
6. The link routine then waits for Adabas to handle the request.
7. Adabas, which is usually running in a different address space or partition, processes the request and posts the link routine, returning all necessary buffers and response codes.
8. The link routine receives control and returns the Adabas buffers with response codes and data to the application.

Available Link Routines

The Adabas API is available for both batch and online applications. For online applications, the Adabas API is under the control of the TP monitor. When Adabas is installed, the Adabas API that is specific to the TP monitor in use is also installed.

For IBM Operating Environments

The following table lists the API versions and the corresponding supported TP monitors available for IBM operating environments:

Link Routine	TP Monitor
ADALCO	Com-plete
ADALNK	Batch (also TSO under OS/390 or z/OS)
ADALNKR	Batch/TSO under OS/390 or z/OS (reentrant)
ADALNA5	Batch for Entire System Server (formerly Natural Process)
ADALNI	IMS/DC
ADALNM	Intercomm
ADALNS	Shadow
ADALTM	Taskmaster
ADALNC	CICS macro-level link. ADALNC is no longer supported for OS/390; it is still supported for VSE/ESA.
LNKOLSC	CICS (command-level)
LNKVSER	Batch under VSE/ESA (reentrant)

For BS2000 Operating Environments

The BS2000 version 7 ADALNK module contains the combined functionality of all the ADALNx modules of previous Adabas versions and contains the following entry points:

Entry Point	TP Monitor
ADALNK	Batch / TIAM
ADALNR	Batch / TIAM (reentrant)
ADALNN	UTM running Natural
ADALNU	UTM with Assembler or a 3GL language
ADALNQ	Adabas SQL Server (ESQ)

Required Work Area

Parameters must be passed to the API. Many of the monitors do not allow standard parameter passing conventions, and the Adabas control block and buffer addresses must be moved into a special work area.

The work area is obtained by the application program from the TP monitor system. It must be specific to the user and addressable by the Adabas API. The application program must initially obtain and clear this area to binary zeros.

Note:

If your application program is written in Natural, the necessary API work area is handled without change to the program code.

The Adabas API versions each have a specific area reserved for use as a work area. For Com-plete, batch/TSO, and IMS/DC, this area is defined by DSECTs provided in the Adabas source library containing the API as follows:

Link Routine	Work Area DSECT	TP Monitor
ADALCO	MODIFLCO	Com-plete
ADALNK / ADALNKR	MODIFIED	Batch/TSO
ADALNI	MODIFLNI	IMS/DC

For these TP monitors, the work area must be at least as long as the DSECT area. The actual required length of this area depends on the TP monitor, and can be determined by examining the assembly listing of the link routine.

For TP monitors with DSECT work areas, a *reentrant* application program must pass to the link routine a pointer to this area as the 7th parameter on the call statement. See *Required Code Reentrancy Properties* for more information.

Once the MODIFxxx area has been initialized, it should not be changed by the application program between Adabas requests.

Work Area for the Batch/TSO Reentrant ADALNKR

Seventh Parameter Required

In addition to the basic six Adabas parameters, the reentrant batch link routine ADALNKR requires a seventh parameter that points to a work area for use by the program on every call. Some Software AG products like XCOM pass eight parameters to ADALNKR.

MODIFIED Work Area

The work area used is mapped by the MODIFIED DSECT, which is provided in the Adabas source library or as an A.book in the VSE sublibrary. The MODIFIED area is at least 192 bytes (decimal) in length. Since the area may be expanded in future Adabas releases, Software AG recommends that you reserve 256 bytes for the area.

Calling Requirements

The work area should be initialized to binary zeros before the first call to ADALNKR, and its contents should not be modified by calling application programs thereafter. Several key fields are stored in this area. If these fields are modified improperly, results are unpredictable and may range from poor performance to abnormal termination of the link routine.

When calling ADALNKR, it is critical to mark the last parameter address in the calling parameter list with an X'80':

- High-level languages do this automatically when their CALL statements are employed.
- Assembler applications can do this by using the CALL macro to invoke ADALNKR.

Required Application Reentrancy Properties

Applications running under most TP monitors must use nonstandard calls to perform functions that are transparently handled by the operating system in a batch environment. In these cases, it is the reentrant properties of application code that determine how multiple users execute Adabas API calls online.

Each Adabas API version complies with the reentrancy requirements for its associated TP monitor. Application programs that use the Adabas API must also comply with the requirements for the TP monitor used.

Note:

The reentrancy requirement set by the TP monitor is a minimum. For example, if the TP monitor requires a quasi-reentrant application program, a fully reentrant program will also be accepted (see CICS special requirement below). However, if a reentrant application program is required, a quasi- or non-reentrant program is not acceptable.

Ideally, code for application programs that are shared by a large number of users (commonly used TP transactions) is "reentrant". The code itself never changes. All work areas are either in general registers or in user-specific work areas that are addressed by general registers. A transfer of control from one user to another requires only a change in the program counter (PSW) and the general registers. Many system routines are coded in this manner.

Most COBOL compilers do not produce reentrant code. The PL/I compiler produces reentrant code, but by using operating system functions that are not allowed by most TP monitors. These limitations have led to the concept of quasi-reentrancy.

A "quasi-reentrant" program may alter its code between calls to TP monitor functions. When a monitor function is invoked, all user data must be saved in a special work area obtained from the TP monitor system. The TP monitor will then schedule another user task as the active task in the system, and this task may reuse the same code. When the original user's task becomes active again, his work area is reestablished and control is passed back to the point at which the user requested a TP monitor function.

The following subsections give more detailed information about the reentrancy requirements of several TP monitors.

Com-plete : Code Reentrancy Requirements

Com-plete does not require nonstandard calling sequences: users may use standard non-reentrant code. Adabas linkage is provided by a Com-plete service routine, which is automatically included in the user's load module if Adabas calls are contained in the user program. The service routine simply passes the user parameters to Com-plete and returns control when the Adabas command has been executed.

CICS : Code Reentrancy Requirements

LNKOLSC, the command-level CICS API, can use either quasi-reentrant or fully reentrant application programs, depending on how the API is installed. Optional installation procedures must be executed in order to use fully reentrant application programs under CICS; however, this allows the use of CICS program isolation under CICS/ESA 4.1.

ADALNC, the macro-level CICS API which is available for VSE/ESA 2.3 environments only, uses quasi-reentrant application programs.

TSO and IMS/DC (Standalone) : Code Reentrancy Requirements

Only TSO and IMS/DC allow the use of "nonreentrant" application programs. However, this is the least efficient means of coding an application program to use with the Adabas API. If a nonreentrant application program is used, it will have to obtain the Adabas communication ID (LTERM, TJID or sign-on ID) on every Adabas request. For TSO and IMS/DC, nonreentrant is the minimum requirement; using a fully reentrant application program will result in significant performance gains (for IMS/ESA, performance gains will result for version 3.1 and above).

Note:

Reentrant and quasi-reentrant application programs must obtain a special work area for the Adabas API. See *Required Adabas API Work Area*.

Adabas Control Block Options

The first parameter passed to the Adabas API by the application program is a pointer to the Adabas control block (ADACB). The ADACB contains information needed to process an Adabas request.

The first byte of the ADACB is used by the Adabas API to determine the processing to be performed. The values for logical requests are:

Hex	Indicates ...
X'00'	a 1-byte file number (file numbers between 1 and 255)
X'30'	a 2-byte file number (file numbers between 1 and 65535)
X'40'	values greater than or equal to a blank. These are accepted as "logical application calls" to maintain compatibility with earlier releases of Adabas. The following calls, however, are reserved for use in special Software AG functions or products and are therefore not accepted: X'44', X'48', and X'4C'.

All other values in the first byte of the ADACB are reserved for use by Software AG.

Using One-Byte File Numbers

For an application program issuing Adabas commands for file numbers between 1 and 255 (single byte), build the control block as follows:

Position	Action
1	Place X'00' in the first byte of the ADACB.
9	Place the file number in the second (rightmost) byte of the ACBFNR field of the ACB. The first (leftmost) byte of the ACBFNR field is used to store the logical (database) ID or number.

If the first byte in ACBFNR is zero, the API will use either the database ID value provided in the DDCARD input data (ADARUN cards) or the default database ID value assembled into the link routine at offset X'80'. Applications written in Software AG's Natural language need not include the first byte of the ADACB because Natural supplies appropriate values.

Using Two-Byte File Numbers

Adabas permits the use of file numbers greater than 255 on logical requests. For an application program issuing Adabas commands for file numbers between 256 and 5000 (two-byte), build the control block as follows:

Position	Action
1	Place X'30' in the first byte of the ADACB.
9	Use both bytes in ACBFNR for the file number, and use the two bytes in ACBRESP for the database (logical) ID.

If the ACBRESP field is zero, the API will use either the database ID from the ADARUN cards provided in DDCARD input data, or the default database ID value assembled into the link routine at offset X'80'.

Using Both One- and Two-Byte File Numbers in a Single Application

Because the application can reset the value in the first byte of the ADACB on each call, it is possible to mix both one- and two-byte file number requests in a single application.

If this method is used, you must ensure the proper construction of the ACBFNR and ACBRESP fields in the ADACB for each call type.

Software AG recommends that an application written to use two-byte file numbers always place X'30' in the first byte of the ADACB, the logical ID in the ACBRESP field, and the file number in the ACBFNR field. The application can then treat both the database ID and file number as 2-byte binary integers, regardless of the value for the file number in use.

Using COBOL to Set the Control Byte

A programming language such as COBOL is not designed to easily manipulate single-byte values as required to establish two-byte file number support for the Adabas API. The following COBOL example illustrates one way to set these values:

```

WORKING-STORAGE SECTION
01  ACB-CONTROL
    05  ACB-TYPE          PIC 9(4) COMP.
    05  ACB-DATA REDEFINES ACB-TYPE.
        07  FILLER        PIC X.
        07  ACB-TYPE-X    PIC X.
01  ADABAS-CB.
    05  ACBTYPE          PIC X.
.
PROCEDURE DIVISION
.
*   FOR SINGLE-BYTE FILE NUMBERS . . .
    MOVE 0 TO ACB-TYPE.
.
*   FOR TWO-BYTE FILE NUMBERS . . .
    MOVE 48 TO ACB-TYPE.
.
    MOVE ACB-TYPE-X TO ACBTYPE.
.
    CALL 'ADABAS' USING ADABAS-CB, . . .
.
.

```

The key to this code segment is the use of the REDEFINES clause to remap the PIC 9(4) COMP field to its constituent two bytes. Then the second byte containing the hexadecimal value for the Adabas control byte can be moved as "character" data to the Adabas control block.

Programming Conventions for Issuing Direct Calls

This section describes the procedures used to issue Adabas calls in direct mode from a program that is to be run under the control of one of the following teleprocessing (TP) monitors:

- Com-plete
- CICS
- IMS/DC
- Shadow II

Com-plete

Application programs that are to be run under control of Com-plete may be coded in exactly the same manner as batch programs. Since each application program is assigned a processing thread by Com-plete, the program need not be written using reentrant or quasi-reentrant code.

The following example shows an Adabas call from a COBOL program that is to be run under Com-plete:

WORKING-STORAGE SECTION

```
.
.
01 CONTROL-BLOCK COPY ADACBCOB.
01 FORMAT-BUFFER COPY FORDEF.
01 RECORD-BUFFER COPY RECDEF.
01 SEARCH-BUFFER COPY SEADEF.
01 VALUE-BUFFER COPY VALDEF.
01 ISN-BUFFER COPY ISNBUF.
```

PROCEDURE DIVISION

```
.
.
CALL 'ADABAS' USING
      CONTROL-BLOCK, FORMAT-BUFFER, RECORD-BUFFER,
      SEARCH-BUFFER, VALUE-BUFFER, ISN-BUFFER.
.
```

CICS

Applications running under VSE CICS version 2.3 may use the macro-level API ADALNC. If the CICS macro-level interface is being used, the work area for the program must be explicitly obtained by

- issuing a call to obtain storage such as:

```
DFHSC TYPE=GETMAIN
```

- making the area addressable by setting the base register (Assembler), loading the BLL cell (COBOL), or setting the pointer variable (PL/ I).

Otherwise, applications running under CICS use the command-level API LNKOLSC and the CICS Transaction Work Area (TWA) to communicate parameters.

The high-level language interface guarantees the quasi-reentrancy of COBOL, PL/ I , and Assembler (release 1.4 and above).

Language	Control blocks and buffers may be defined ...
COBOL	in working storage. All of working storage is copied to a user work area when a transaction is initiated.
PL/ I	as automatic storage (default storage class) variables.

The addresses of the Adabas control block and buffers are passed in the same way for all releases of CICS. These addresses must be placed in the first six words of the TWA.

To place the parameter addresses in the TWA, Software AG provides an Assembler subroutine that can be called from a COBOL or Assembler application program. The subroutine uses entry point ADASTWA and accepts the TWA as its first parameter.

The 'EXEC CICS ADDRESS TWA' function is used to make the TWA addressable. The second to seventh parameters are the usual Adabas calling parameters. The Assembler subroutine places the parameter addresses into the TWA, and the CICS/Adabas link routine retrieves them from the TWA.

Addressing the CICS TWA : Assembler

A CICS Assembler programmer can address the TWA directly by using an installation macro to place the addresses in the TWA and call Adabas.

Addressing the CICS TWA : PL/I

PL/I offers a facility for addressing the TWA and obtaining the addresses of data areas. The programmer himself can place parameter addresses in the TWA. Your site may wish to establish a PL/I preprocessor procedure to generate the calling code.

```
DCL 1          TWA  BASED  (TWAPTR) ,
      2      CBPTR  POINTER ,
      2      FBPTR  POINTER ,
      2      RBPTR  POINTER ,
      .
      EXEC  CICS  ADDRESS  TWA  (TWAPTR)  END-EXEC ;
      .
      .
      CBPTR=ADDR ( ADA-CONTROL-BLOCK ) ;
      FBPTR=ADDR ( FORMAT-BUFFER ) ;
      RBPTR=ADDR ( RECORD-BUFFER ) ;
```

Adabas Call Using CICS : VS COBOL

Under VS COBOL, Adabas is called using the statement:

```
EXEC CICS LINK PROGRAM ('ADABAS')
      END-EXEC.

CBL XOPTS (APOST)
IDENTIFICATION DIVISION.
.
.
WORKING-STORAGE SECTION.
.
.
01  ADABAS-CB    COPY  ADACBCOB.
01  ADABAS-FB    COPY  ADAFBCOB.
01  ADABAS-RB    COPY  ADARBCOB.
01  ADABAS-SB    COPY  ADASBCOB.
01  ADABAS-VB    COPY  ADAVBCOB.
01  ADABAS-IB    COPY  ADAIBCOB.
.
.

LINKAGE SECTION.
.
.
01  PARMLIST.
      05  FILLER      PIC S9(08) COMP.
      05  TWAPTR      PIC S9(08) COMP.

01  TWA.
      05  PARM-ADDRESSES OCCURS 7 TIMES      PIC S9(08) COMP.
.
.
PROCEDURE DIVISION.
.
.
```

```

SERVICE RELOAD PARMLIST.
.
EXEC CICS ADDRESS TWA (TWAPTR) END-EXEC
SERVICE RELOAD TWA.
.
CALL 'ADASTWA' USING TWA, ADABAS-CB, ADABAS-FB,
                        ADABAS-RB, ADABAS-SB, ADABAS-VB
                        ADABAS-IB.
EXEC CICS LINK PROGRAM ('ADABAS') END-EXEC.
.
.
.

```

Adabas Call Using CICS : COBOL II or COBOL/LE

Under COBOL II or COBOL/LE, Adabas is called using the statement:

```
EXEC CICS LINK PROGRAM ('ADABAS') END-EXEC.
```

```

CBL XOPTS (APOST,ANSI85)
IDENTIFICATION DIVISION.
.
.
WORKING-STORAGE SECTION.
.
.
01  ADABAS-CB    COPY  ADACBCOB.
01  ADABAS-FB    COPY  ADAFBCOB.
01  ADABAS-RB    COPY  ADARBCOB.
01  ADABAS-SB    COPY  ADASBCOB.
01  ADABAS-VB    COPY  ADAVBCOB.
01  ADABAS-IB    COPY  ADAIBCOB.
.
.
LINKAGE SECTION.
.
01  TWA.
    05  PARM-ADDRESSES OCCURS 7 TIMES    PIC S9(08) COMP.
.
.
PROCEDURE DIVISION.
.
.
.  EXEC CICS ADDRESS TWA (ADDRESS OF TWA) END-EXEC.
.
CALL 'ADASTWA' USING TWA, ADABAS-CB, ADABAS-FB,
                        ADABAS-RB, ADABAS-SB, ADABAS-VB,
                        ADABAS-IB.
EXEC CICS LINK PROGRAM ('ADABAS') END-EXEC.
.
.
.

```

IMS/DC

IMS message processing programs that use the Adabas API require no special link and need not be reentrant. However, a reentrant option is supported: the application code and the Adabas IMS API module ADALNI can function with full reentrancy if the application program provides a work area as the 7th parameter when calling the API (see example below).

Under IMS/ESA 3.1 and above, the API ADALNI should be linked with an AMODE of 31 because the IMS control blocks referenced by the routine may be above the 16-megabyte line.

Adabas Call Using IMS/DC (Nonreentrant)

A nonreentrant Adabas API call under IMS/DC is made like a conventional Adabas API call under batch as follows:

```
WORKING-STORAGE-SECTION.
.
.
01  ADA-CONTROL BLOCK   COPY  ADACBCOB.
01  FORMAT-BUFFER      COPY  FORDEF.
01  RECORD-BUFFER      COPY  RECDEF.
01  SEARCH-BUFFER      COPY  SEADF.
01  VALUE-BUFFER       COPY  VALDEF.
01  ISN-BUFFER         COPY  ISNDEF.
.
PROCEDURE DIVISION.
.
.
    CALL 'ADABAS' USING ADA-CONTROL-BLOCK, FORMAT-BUFFER,
                        RECORD-BUFFER, SEARCH-BUFFER,
                        VALUE-BUFFER, ISN-BUFFER.
.
.
```

Adabas Call Using IMS/DC (Reentrant)

The Adabas ADALNI module can be assembled to be functionally reentrant. Refer to the Adabas Installation documentation for information about setting up ADALNI as a reentrant module.

To facilitate reentrant operation, an additional parameter pointing to a work area obtained by the caller must be passed to the ADALNI routine. This area must be initialized to binary zeros before the first call to the Adabas IMS API, and must not be modified between calls. The length of the ADALNI reentrant work area can be determined by examining the current assembly listing. Under Adabas, the length is 128 bytes.

The reentrant Adabas API call under IMS/DC is as follows:

```
WORKING-STORAGE-SECTION.
.
.
01  ADA-CONTROL BLOCK   COPY  ADACBCOB.
01  FORMAT-BUFFER      COPY  FORDEF.
01  RECORD-BUFFER      COPY  RECDEF.
01  SEARCH-BUFFER      COPY  SEADF.
01  VALUE-BUFFER       COPY  VALDEF.
01  ISN-BUFFER         COPY  ISNDEF.
01  ADALNI-WORK-AREA   PIC  X(128).
.
PROCEDURE DIVISION.
    MOVE LOW-VALUES TO ADALNI-WORK-AREA.
.
.
    CALL 'ADABAS' USING ADA-CONTROL-BLOCK, FORMAT-BUFFER,
                        RECORD-BUFFER, SEARCH-BUFFER,
```

```

VALUE-BUFFER, ISN-BUFFER.
ADALNI-WORK-AREA.

```

Shadow II

The following procedure is used for calling Adabas from quasi-reentrant COBOL application programs that are to be run under Shadow II:

1. Define storage for variable Adabas control block and buffers in the linkage section of the COBOL program. Obtain storage using the Shadow get-variable function, ISHDHLGV.
2. Pass parameters to the Adabas interface routine directly using the Shadow calling procedure. Include in the call the count of the number of parameters as a binary halfword number.

The Adabas interface routine is linked as part of the Shadow monitor and is defined in the Shadow program table.

3. Use the Shadow routine ISHDHLCF to access the Adabas interface routine. The first parameter is the character name of the interface routine (ADABAS); the second, the number of parameters being passed to the routine; followed by the parameters to be passed to Adabas.

Adabas Call Using Shadow II

```

WORKING-STORAGE SECTION.
.
.
01  ADABAS          PIC X(8) VALUE 'ADABAS'.
.
.
01  PARM-NUMBER     PIC S9(4) COMP VALUE +6.
01  ISHD-H80        PIC S9(4) COMP VALUE +80.
.
.
LINKAGE-SECTION.
.
.
01  ADABAS-CONTROL-BLOCK  COPY ADACBCOB.
.
.
PROCEDURE DIVISION USING TCBD RELOCATE.
.
.
GET-STORAGE.
CALL  'ISHDHLGV' USING ISHD-FULLWORD0, ISHD-XF0,
      ISHD-H80, BLL-NUMBER.
.
.
.
CALL-ADABAS.
      CALL 'ISHDHLCF' USING      ADABAS, PARM-NUMBER,
                                ADABAS-CONTROL-BLOCK,
                                FORMAT-BUFFER, RECORD-BUFFER,
                                SEARCH-BUFFER, VALUE-BUFFER,
                                ISN-BUFFER.
.
.

```

Using the Adabas API in Batch Mode

The Adabas API in batch mode uses a standard call with a parameter list in register 1 and register 13 pointing to a register save area. This convention is supported by all major programming languages through their CALL mechanisms.

Under most mainframe operating systems, the batch API (ADALNK) can either be linked directly with the batch application module or it can be loaded by ADAUSER. Software AG *strongly* recommends that batch applications be linked with ADAUSER and not the batch API (ADALNK).

ADAUSER and ADARUN with the Adabas API

The ADAUSER module can optionally be linked with the Adabas API. ADAUSER provides upward compatibility with Adabas releases and a degree of isolation from changes that might be made in the API or the Adabas SVC in the future.

Each user program to be executed should be linked with the Adabas version-independent module ADAUSER, which dynamically loads the Adabas control module ADARUN. For batch mode execution, the user program should be linked with ADAUSER to achieve maximum environment independence, as shown below:

User Program linked with ...	Advantage
ADAUSER, ADARUN, and ADALNK	independent of mode and Adabas version
ADARUN and ADALNK	independent of mode only
ADALNK	none; no version- or mode-independence

The following sections illustrate the JCL/JCS required to link the batch application module with ADAUSER.

Link Example (BS2000)

```
/ EXEC $TSOSLNK
PROGRAM USERPROG
INCLUDE USERPGM,           ... User Library
INCLUDE ADAUSER,          ... Adabas Library
END
```

Link Example (OS/390 or z/OS)

```
// EXEC LKED,PARM='NCAL'
//LKED.SYSLMOD DD           ... User Library
//LKED.ADALIB DD           ... Adabas Library
//LKED.SYSIN DD *
    INCLUDE SYSLMOD(USERPGM)
    INCLUDE ADALIB(ADAUSER)
    ENTRY USEREP           (see note)
    NAME USERPROG(R)
/*
```

Note:

The entry point, if specified, must be the entry point of the user program.

Link Example (VM/ESA or z/VM)

```
FILEDEF ADALIB DISK ADAVnnn LOADLIB fm
FILEDEF SYSLIN DISK LINKEDIT CARDS A
LKED userprog ( NCAL LET XREF MAP LIBE USERLIB LIST
```

The file "LINKEDIT CARDS A" contains the following linkage editor control statements. The user program exists as a TEXT file.

```
INCLUDE ADALIB(ADABAS)
ENTRY userprog
NAME userprog(R)
```

Note:

Link with ADAUSER is not applicable when using CMS since this system dynamically loads all necessary modules.

Link Example (VSE/ESA)

```
* Appropriate assignments must be made for private
libraries, where necessary.
*
// OPTION CATAL
  PHASE USERPROG,*
  INCLUDE USERPGM
  INCLUDE ADAUSER
  ENTRY USEREP                (see note)
// EXEC LNKEDT
```

Note:

The entry point, if specified, must be the entry point of the user program.

Batch Execution Modes

When executing under batch, the program can be run in either single-user or multiuser mode:

- Single-user mode runs the application program, the batch API, ADARUN, and the Adabas nucleus in the same address space or partition.
- Multiuser mode executes the application program and the Adabas API in an address space separate from the Adabas nucleus.

The recommended mode of operation is multiuser mode. The user must provide only those job control statements required by ADARUN and the user program.

Multiuser Mode Example (BS2000)

In SDF Format:

```
/ASS-SYSDTA *SYSCMD          (ADARUN PARAMETERS)
/SET-FILE-LINK DDLIB,ADAvrs.MOD
/SET-FILE-LINK BLSLIB00,user modlib
/START-PROGRAM USERPROG,PR-MO=ANY,RUN-MODE=ADV(ALT-LIB=YES)
ADARUN MODE=MULTI...
```

In ISP Format:

```
/FILE ADABAS MODLIB, LINK=DDLIB
/SYSFILE TASKLIB=user modlib
/SYSFILE SYSDTA=(SYSCMD) (ADARUN PARAMETERS)
/EXEC USERPROG
ADARUN MODE=MULTI...
```

Notes:

1. As an alternative to using SYSDTA as the input stream, the user program can assign a sequential file containing the ADARUN parameters to the link name DDCARD using /SET-FILE-LINK (in ISP format, /FILE).
2. Software AG recommends that you link the ADAUSER module to user programs in a TP environment; for example, COBOL.

Multiuser Mode Example (OS/390 or z/OS)

The following example assumes that the user program USERPROG has been linked with the module ADAUSER and is to be executed in multiuser mode.

```
// EXEC PGM=USERPROG
//STEPLIB DD ... User Library
// DD ... Adabas Library
//DDCARD DD *
ADARUN MODE=MULTI
//DDPRINT DD SYSOUT=*
//... user DD statements ...
```

Multiuser Mode Example (VM/ESA or z/VM)

User programs running under VM/ESA or z/VM in multiuser mode cannot use ADARUN to control other programs.

Multiuser Mode Example (VSE/ESA)

The following example assumes that the user program USERPROG has been linked with the module ADAUSER and is to be executed in multiuser mode.

```
//...user program assignments...
// LIBDEF PHASE,SEARCH=(user-library, ADABAS-library)
// EXEC USERPROG
ADARUN MODE=MULTI
/*
```

If the user program reads statement input, one of the following applies:

- If all user statements are read before the first Adabas call, they must immediately follow the EXEC statement and be followed by /*. The user file must be opened, read, and closed before the first Adabas call.
- If the first Adabas call is made before the first user statement is read, the user statements must follow the ADARUN parameter statements and start with a /* statement.

- Otherwise, the ADARUN parameter statements must be read from file CARD on tape or disk.

Notes:

1. ADARUN and/or the Adalink, ADAIOR, ADAMOD, and other Adabas modules must be available for dynamic loading during execution.
2. The Adabas load library must be available during execution so that required modules can be dynamically loaded.

Execution in Single-User Mode

In single-user mode, the appropriate Adabas nucleus JCL must be included with the JCL of the user program. This includes job control statements to define the Adabas datasets for the Associator, Data Storage, the Work dataset, and any datasets for protection or command logging. For more information about Adabas runtime job control requirements, see *Adabas Session Execution*.

Note:

User programs that use VM/ESA or z/VM facilities cannot access an Adabas nucleus in single-user mode.

Support for OpenEdition OS/390 Adabas Clients

A client running under OpenEdition OS/390 or z/OS can access Adabas. An OpenEdition application containing calls to Adabas can be linked with ADALNK (option 1) or ADAUSER (option 2).

Software AG recommends that you link your OpenEdition application with ADAUSER (option 2) for the following reasons:

- the application is not tied to a specific database ID and SVC number, or Adabas release;
- the DDPRINT output provides information about the database ID and SVC number used, as well as diagnostic information in case of error (DDPRINT output is lost when using option 1); and
- the program occupies less space in the hierarchical file system (HFS).

Option 1 : Link OpenEdition Application with ADALNK

An OpenEdition application that contains calls to Adabas can be linked with the module ADALNK. The database ID and SVC number must be zapped into the Adabas CSECT of the linked module at the offsets described in section *Writing User Exits for an Adalink* of the Adabas Installation documentation.

The following sample ZAP and link job has the following steps:

- COPYLNK: copy module ADALNK to another library
- ZAPLNK: zap the copied ADALNK module
- BINDAPP1: link (bind) the application with the zapped ADALNK into OpenEdition

```
// *
// * COPY AND RENAME ADALNK
// *
//
COPYLNK
```

```

EXEC PGM=IEBCOPY
//INLIB DD DSN=ADABAS.load.library,DISP=SHR
//OTLIB DD DSN=ADABAS.lnk.library,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
COPY INDD=INLIB,OUTDD=OTLIB
SELECT MEMBER=(( ADALNK,ADALNKOE,R))
/*
/**
/** ZAP DBID AND SVC INTO COPIED ADALNK
/**
//
ZAPLNK

EXEC PGM=IMASPZAP
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DSN=ADABAS.lnk.library,DISP=SHR
//SYSIN DD *
NAME ADALNKOE ADABAS
VER 0080 0001 DEFAULT DBID 1
VER 0084 0AF9 DEFAULT SVC 249
REP 0080 00D3 <===== CHANGE TO USER DBID (HERE DBID 211)
REP 0084 0AE8 <===== CHANGE TO USER SVC (HERE SVC 232)
/*
/**
/** BIND APPLICATION
/**
//
BINDAPP1

EXEC PGM=IEWBLINK,
// PARM='LIST,LET,XREF,MAP,CASE=MIXED'
//SYSPRINT DD SYSOUT=*
//SYSLMOD DD PATH='/u/group/user',
// PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
// PATHMODE=(SIRWXU,SIRWXG,SIRWXO)
//APPLIB DD DSN=your.appl.library,DISP=SHR
//LNKLOAD DD DSN=ADABAS.lnk.library,DISP=SHR
//SYSLIN DD *
INCLUDE APPLIB(applname)
INCLUDE LNKLOAD(ADALNKOE)
ENTRY applent
NAME oeappl(R)
/*

```

Option 2 : Link OpenEdition Application with ADAUSER

An OpenEdition application that contains calls to Adabas can be linked with the module ADAUSER.

Additionally, a member "ddcard" must be set up in the OpenEdition hierarchical file system (HFS) to contain the ADARUN parameters required by the client; for example:

```
ADARUN PROG=USER,DBID=211,SVC=232,MODE=MULTI
```

Prior to the first call to Adabas, the application must set the current working directory (using the `chdir()` function, for example) to the directory where file "ddcard" is located. As the application runs, Adabas searches the current working directory for member "ddcard", and extracts the parameters. Additionally, Adabas directs the DDPRINT output to member "ddprint" of the current working directory.

Note:

Member names "ddcard" and "ddprint" are case-sensitive. Member name "DDCARD" is not valid and will be ignored.

The following sample link job has one step:

- BINDAPP2: link (bind) the application with ADAUSER into OpenEdition

```

/*
/* BIND APPLICATION
/*
//
BINDAPP2

EXEC PGM=IEWBLINK,
//          PARM='LIST,LET,XREF,MAP,CASE=MIXED'
//SYSPRINT DD  SYSOUT=*
//SYSLMOD  DD  PATH='/u/group/user',
//          PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//          PATHMODE=(SIRWXU,SIRWXG,SIRWXO)
//APPLIB   DD  DSN=your.appl.library,DISP=SHR
//ADALOAD  DD  DSN=ADABAS.load.library,DISP=SHR
//SYSLIN   DD  *
INCLUDE APPLIB(applname)
INCLUDE ADALOAD(ADAUSER)
ENTRY  applent
NAME  oeappl(R)
/*

```

Setting the OpenEdition Shell Variable STEPLIB

For both options, the OpenEdition shell variable STEPLIB must be set to ensure access to the Adabas load library. The following sample job sets the variable from OpenEdition running in batch mode:

```

/*
//OEBATCH EXEC PGM=BPXBATCH,
//          PARM='PGM /u/group/user/oeappl'
//STDIN   DD
PATH='/u/group/user/oeappl.in',PATHOPTS=(ORDONLY)
//STDOUT  DD  PATH='/u/group/user/oeappl.out',
//          PATHOPTS=(OWRONLY,OCREAT),PATHMODE=SIRWXU
//STERR   DD  PATH='/u/group/user/oeappl.err',
//          PATHOPTS=(OWRONLY,OCREAT),PATHMODE=SIRWXU
//STDENV  DD  *
STEPLIB=ADABAS.load.library
/*
//

```

Limitations for OpenEdition Support

Support is *not* available for running the following under OpenEdition:

- the Adabas nucleus or utilities
- clients running in single-user mode (MODE=SINGLE)

- clients running in 24-bit addressing mode (AMODE 24)

Adabas Restart and Recovery

A "user session" is a sequence of Adabas calls optionally starting with an OP command and ending with a CL command. A "user" is either a batch mode program or a person using a terminal. The user can be identified by an 8-byte unique ID provided with the OP command. This ID enables Adabas to retain restart information (ET data) beyond the end of a session.

For communication purposes, a terminal user is identified by machine, address space, and terminal ID, thereby ensuring that each user ID is unique.

During an Adabas "session" (from activation to termination), the Adabas nucleus creates a sequence of protection entries in exact historical sequence reflecting all modifications made in the database. The sequence of protection entries is written to the Work dataset (part 1) and to a protection log in the form of blocks. Each block contains the nucleus session number, a unique block number, and a time stamp.

This chapter covers the following topics:

- Work Dataset
 - Protection Log
 - Restart Operations
 - Database Recovery
-

Work Dataset

Part 1 of the Work dataset (ADARUN LP parameter) stores the most recent protection entries using a wrap-around method.

The protection entries on the Work dataset are used to execute a BT command and to execute autorestart/autobackout when reactivating the Adabas nucleus after a system failure. The entries may contain all or some of the following:

- before/after images of a data record;
- before/after images of elements of the inverted lists (DVT);
- special before-images of inverted list blocks for automatically repairing the database after a system failure;
- checkpoint entries;
- ET entries including ET data;
- special entries for handling the internal restart procedures.

Adabas identifies a batch user by checking a store clock (STCK) value during the program's first Adabas call.

Protection Log

The protection log contains the same entries as part 1 of the Work dataset (except the special before-images mentioned in the discussion of the Work dataset. Additional entries on the protection log which are not stored on the Work dataset include an entry for the infrequently used C5 data and an after-image Associator or Data Storage block written during buffer flush; the latter occurs while an online SAVE function of the ADASAV utility is running.

All protection log entries created by the nucleus describe the modifications made to the database in exact historical order. Each block is associated with a block sequence number.

The protection log can be written either

- directly to a sequential protection log (DD/SIBA) dataset; or
- to multiple protection log (DD/PLOGR1, DD/PLOGR2, ... DD/PLOGR8) datasets.

Note:

Adabas version 7.4 still supports dual protection logging using DUALPLD/S and user exit 2.

Multiple protection log datasets are each physical datasets of the same size and block length, randomly accessed, and used in succession. This means that one of the datasets can be used and written by the nucleus while others are being copied in order for archive purposes.

Sequential Protection Log

The sequential protection log dataset is opened by the nucleus when a session is started and closed when a session is ended. In general, this dataset is assigned to tape to avoid disk space problems that may cause an unexpected abnormal termination. Because the amount of data to be written to such a dataset depends upon the amount of activity of all users, estimating disk space is difficult.

At the end of the Adabas session, an end-of-file mark (EOF) is written to the tape to indicate end of session. Adabas supports multivolume protection log datasets. If one reel is not enough to store all protection log entries, subsequent tape reels can be used.

The nucleus writes a checkpoint for each volume written. This checkpoint contains information about session number, volume serial number, and block sequence numbers.

The ADARES COPY function must be used to copy such a sequential protection log dataset for archive purposes.

If a nucleus session ended abnormally, the final EOF mark on the tape may not be written. This may cause problems if the protection log is used directly as input to the ADARES BACKOUT or REGENERATE function. The COPY function of ADARES is able to detect the logical end of the in/out protection log and writes a valid EOF mark to the output. ADARES writes checkpoints in the same way the nucleus does for each output volume.

Note:

When using a sequential protection log on tape, one tape unit is allocated for the Adabas nucleus for the duration of the entire nucleus session. This session can last a very long time, during which the log tape unit must be available; a single sequential logging tape is therefore not adequate in every case. Software AG therefore recommends using multiple protection logging.

Multiple Dataset Protection Log

The Adabas multiple dataset protection log consists of two to eight datasets (DD/PLOGRn where "n" is the sequential number of the dataset) with the following attributes:

- fixed block size;
- reside on DASD;
- preformatted by ADAFRM;
- all datasets have the same number of blocks and identical block sizes;
- all datasets can be shared by the nucleus and other utilities (ADARES).

Assuming newly formatted dual or multiple protection log datasets, Adabas selects DD/PLOGR1 at startup and starts writing protection log entries to it. Writing starts at block 2. Block 1 contains status information about the dataset. Other PLOG datasets are still unused. Protection log entries are written to multiple protection log datasets in the same order they are written to a sequential log.

Each protection log dataset need not be large enough to accommodate all protection log entries for a session. When one dataset becomes full, "protection log switching" occurs as follows:

1. status information is written to block 1 to terminate the current protection log dataset;
2. there is a switch to another dataset;
3. a message is written to the operator and to the log output; and
4. user exit 12 is called (see below).

While the nucleus continues writing protection log entries to the other dataset, the first one is copied to a sequential dataset by ADARES PLCOPY. ADARES can be started manually or initiated by user exit 12, which is called whenever a switch from one protection log dataset to another occurs. ADARES writes a checkpoint for each output volume written. This checkpoint contains the session number, volume serial number, and block sequence number.

A protection log switch may occur more than once in a single session. The content of each protection log dataset must be copied to a single sequential dataset. All subsequent copies produced within one session are logically equivalent to the information the nucleus would have written to a sequential protection log (DD/SIBA).

All sequential copies can be concatenated to form a single sequential dataset containing all protection log entries for a session. In fact, a sequential copy is required as input by the BACKOUT/REGENERATE functions of ADARES.

Note:

A tape unit to store the sequential protection log entries is required only during the ADARES PLCOPY run.

If multiple dataset protection logging is used, but user exit 12 is not available to call ADARES PLCOPY, protection log switching occurs as follows:

1. the current protection log dataset is closed; and
2. if no other dataset is empty, the following message is issued and the old data is overwritten:

Now it's too late to copy DDPLOGRn (or PLOGRn)

In this case, protection log information is lost.

Restart Operations

Protection entries are needed if any of the following fail:

- a user application program
- Adabas
- the operating system
- the hardware

Restart after a User Application Program Failure

An application program that is in the middle of a transaction can detect that the transaction cannot be completed successfully. Removing the first portion of the transaction, called back out or roll back, is performed by the BT command.

The BT command is executed by reading the Work dataset backwards and executing the entries for the specific transaction in reverse (after-image is used to scratch an element in the database, before-image is used to insert an element in the database). The start-transaction bit in an element serves as the stop indicator for the BT process.

Restart after an Adabas, Operating System, or Hardware Failure

When Adabas is reactivated after any failure that caused the Adabas nucleus to terminate abnormally (that is, failure of Adabas, the operating system, or hardware), an automatic procedure is executed to bring the database to a physically and logically valid status. All partially executed update commands are reset. All incomplete transactions are backed out.

This automatic procedure comprises three steps:

1. repair the database
2. autostart
3. autobackout

The repair is needed to modify the database to the status it would have if a buffer flush had just been completed at the time of the failure. In other words, all blocks in the database are at a status that enables the nucleus to perform normally by addressing Data Storage records through the address converter and normal index entries through the upper index.

"Autorestart" backs out updates of single update commands that were partially executed when the system failed; "Autobackout" backs out updates of user transactions that were partially executed when the system failed.

The major protection entries used for autorestart and autobackout are the before-images and after-images of Data Storage and the inverted lists (DVT).

Restart after a Power Failure

Depending on the hardware, a power failure during an I/O operation may damage the Adabas blocks that were being processed. This damage cannot be detected during autorestart and therefore can result in problems later, such as unexpected response codes of lost database updates.

Note:

If the cause of the ABEND was a power failure, Software AG strongly recommends recovering the affected files using the ADASAV and ADARES utilities as described in the section *Database Recovery*.

Whenever an Adabas session is reactivated with the IGNDIB=YES parameter, which forces the new session to ignore an existing session communication block (DIB) in the Associator, Adabas checks whether a buffer flush was active when the ABEND occurred. If a buffer flush was in process, the autorestart shuts down and issues an ADAN58 message:

ADAN58 BUFFER-FLUSH START RECORD DETECTED DURING AUTORESTART. THE NUCLEUS WILL TERMINATE AFTER AUTORESTART. IN CASE OF POWER FAILURE, THE DATABASE MIGHT BE INCONSISTENT...

The message also includes a list of the files that were being updated when the buffer flush was in process. In this case, the DBA must check whether the cause of the ABEND was a power failure.

If the ABEND was *definitely not* a power failure and the integrity of the information on the output hardware can be guaranteed, the database can be reactivated immediately. Database recovery is not necessary.

Using Automatic Restart Management (ARM)

Automatic restart management (ARM) is used to automatically restart a nucleus when it ABENDs. Automatic restart is suppressed when the ABEND is intentional; for example, when it results from a parameter error.

ARM can be used for Adabas nuclei in both cluster and non-cluster environments.

The ADARUN parameter ARMNAME is used to identify the element in the ARM 'policy' that is to be activated. Each element specifies when, where, and how often an automatic restart is to be attempted. If an ARM policy has not been defined, the ARMNAME parameter has no effect.

Database Recovery

If an Adabas, operating system, or hardware failure occurs and the physical database is still readable (which is the normal case), the Adabas nucleus automatically takes all necessary steps to ensure that database processing can be continued in a normal manner.

Database recovery is described under the following headings:

- Recreating a Database
- Database or File Recovery Considerations
- Database Recovery Guidelines
- Using the Adabas Recovery Aid (ADARAI)

Recreating a Database

In case of a head crash (physical damage to the database), application program error (logical damage to the database), or a power failure during a buffer flush (described in the section *Restart after a Power Failure*, the utilities ADASAV and ADARES must be used to recreate the database.

▶ To restore and regenerate the entire database

1. Restore the database with ADASAV using the sequential dataset containing the most recent copy of the database.
2. Restore the database blocks that were updated during the SAVE of the database (online SAVE).
3. Regenerate with ADARES from the checkpoint taken at the end of the SAVE function to the latest point at which the database was still intact (this is done automatically by Adabas).

▶ Files re-creating after failure To restore and regenerate a single file of the database

- Use the steps described for restoring and regenerating an entire database, but make the appropriate changes in the parameter statements of the utilities.

▶ To restore and regenerate single blocks of Data Storage

- Use the REPAIR function of ADARES.

▶ To restore the status of a database at the start of a single batch update run

1. If a long-running batch program performs a large number of erroneous updates (e.g., logic error in program), and the program was the only user performing updates, it may be desirable to back out all updates performed by the program.
2. Use the ADARES BACKOUT function with the sequential, dula, or multiple protection log.

Database or File Recovery Considerations

The information written on the protection log (SIBA/PLOG) contains record-based information, including the record identifiers (ISNs). The ADARES utility passes the ISN-based information to Adabas; however, Adabas can neither verify the validity of the ISN nor check that the logical content of the record is correct.

Therefore, you must ensure that the combinations of ISN and logical record used in the original session are also used during the procedure for recreating the file or database. Use the ISNs and parameters that were specified when the file was originally loaded.

Database Recovery Guidelines

To ensure that the database can be recovered in the event of a software or hardware failure, Software AG recommends that the DBA

- create a sequential dataset copy of the database using the ADASAV utility (SAVE function) and archive the output dataset (normally on tape). This dataset reflects the status of the database at a specific time.
- retain all protection log data written to the protection log dataset for each nucleus session.

If you are using sequential protection logging (SIBA), you can use the ADARES COPY function to archive the log information. If you are using multiple dataset protection logging (PLOG), use the ADARES PLCOPY function.

Each nucleus session is identified by a unique session number. This number is assigned to all SAVE and protection log data.

Example 1: Inactive Nucleus Save Operation

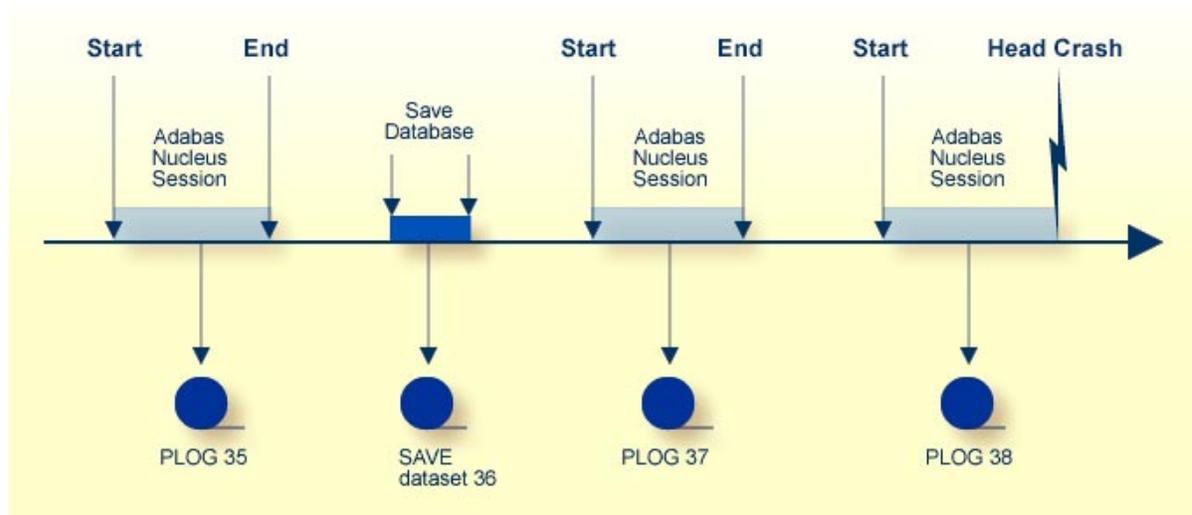
► To recreate the database to the status of the failure

1. Restore the database using SAVE dataset 36.

With the nucleus *not* active, execute

```
ADASAV RESTORE
```

2. Start the Adabas nucleus.



3. Reapply all modifications made in session 37.

With the nucleus active and running, execute

```
ADARES REGENERATE PLOGNUM=37
```

4. Reapply all modifications made in session 38.

With the nucleus active and running, execute

```
ADARES REGENERATE PLOGNUM=38
```

Example 2: Active Nucleus Save Operation

This example shows the assignment of session numbers to SAVE and protection log datasets when a SAVE database function is executed parallel to an active nucleus.

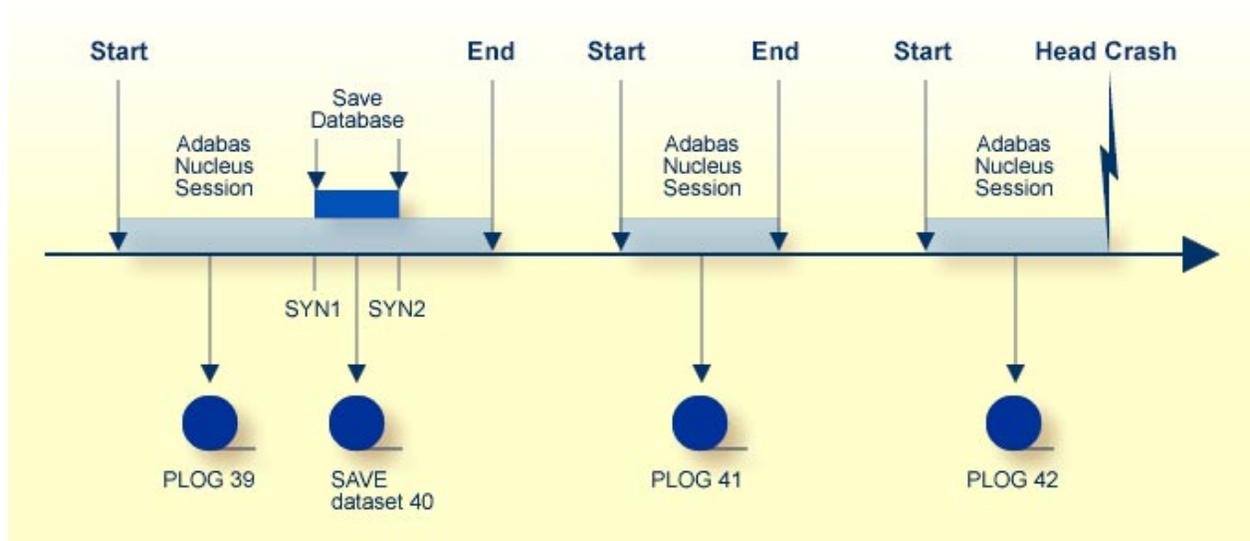
The nucleus writes a protection log number 39.

When ADASAV starts the SAVE function, the nucleus writes a SYN1 checkpoint to the data protection log.

At the end of the SAVE function, a SYN2 checkpoint is written. The SYN2 checkpoint is a synchronized checkpoint where all users are at ET status.

All writes of the nucleus to the Associator and Data Storage are written to the protection log as well (between SYN1 and SYN2).

The output of ADASAV is associated with session number 40. The nucleus still writes to protection log 39, even if the SYN2 checkpoint has been written.



► To reset the database to the status of the failure

1. Restore the database using SAVE dataset 40 as input.

Modifications made to the database during the online SAVE operation are found on protection log 39, starting at the SYN1 checkpoint. ADASAV also requires the correct protection log input.

2. Stop the nucleus; then execute

```
ADASAV RESTONL PLOGNUM=39 , SYN1=blocknumber
```

3. Start the Adabas nucleus.

4. Reapply modifications made in session 39, starting from the SYN2 checkpoint.

With the nucleus active, execute

```
ADARES REGENERATE PLOGNUM=39 , FROMCP=SYN2 , FROMBLK=blocknumber
```

5. Reapply modifications made in session 41.

With the nucleus active, execute

```
ADARES REGENERATE PLOGNUM=41
```

6. Reapply modifications made in session 42 up to the time of failure.

With the nucleus active, execute

```
ADARES REGENERATE PLOGNUM=42
```

To find the SYN1 and SYN2 block numbers needed as input for ADASAV and ADARES, check the output report written by the ADASAV SAVE function or perform the "ET Checkpoint" function in Adabas Online System.

Using the Adabas Recovery Aid (ADARAI)

"Transaction" recovery is provided whenever an Adabas session is abnormally terminated. The Adabas autobackout routine, which is automatically invoked at the beginning of every Adabas session, removes the effects of all interrupted transactions from the database.

However, when a database dataset (ASSO, DATA, or WORK) is destroyed, it is necessary to restore and regenerate the database to recover the lost data.

The Adabas Recovery Aid helps automate and optimize "database" recovery. It records and reports all information needed to recover the database and builds the recovery job stream (JCL/JCS), which is the basis for reexecuting the jobs performed from the time of the last SAVE to the point of failure and error. For information, see the description of the ADARAI utility in the Adabas Utilities documentation.

Note:

The job stream generation function is not yet available under VSE/ESA or VM/ESA.

Adabas Timeout Feature

Each user who calls Adabas is assigned a user queue element (UQE) that indicates the nucleus resources allocated to the user. If a user calls Adabas and is then inactive for an extended period (for example, because a terminal operator leaves the terminal without logging off), that user's resources are locked and are not available to other users.

Adabas provides a timeout mechanism which prevents this potential waste of resources. The Adabas timeout function allows control of session-wide or individual user time limits for overall transaction time, and for specific user types.

This chapter covers the following topics:

- Timeout Parameters
- Action Taken If Time Limit Exceeded

Timeout Parameters

The following ADARUN timeout parameters can be set at nucleus startup:

Parameter	Specifies the maximum elapsed time (in units of 1.048576 seconds) ...
TNAA	an access-only user may be active without issuing an Adabas call.
TNAE	an ET-logic user may be active without issuing an Adabas call.
TNAX	an exclusive-control user may be active without issuing an Adabas call.
TT	permitted for a logical transaction by an ET-logic user.

If not specified, a timeout parameter value defaults to 900 or slightly over 15 minutes.

These parameter values, which apply to all users in the session, can be overridden for a single user with an Adabas OP command for the user session. The maximum values for such OP command overrides are specified by the following ADARUN timeout parameters set at nucleus startup:

Parameter	Specifies the maximum elapsed time (in units of 1.048576 seconds) ...
MXTNA	any individual user may be active without issuing an Adabas call.
MXTT	permitted for a logical transaction by an individual ET-logic user.

The Adabas ADADBS OPERCOM utility or Adabas Online System may also be used to change the timeout values.

The following section describes the specific actions taken for each type of timeout.

Action Taken If Time Limit Exceeded

This section lists the actions the nucleus may take when a time limit expires or the STOPU command is issued and describes the circumstances that determine the action.

Notes:

1. The DELUI/DELUF operator commands, the ADADBS OPERCOM STOPI/STOPF commands with the PURGE option, or their Adabas Online System (AOS) counterparts are handled like STOPU.
2. The STOPI/STOPF operator commands, the ADADBS OPERCOM STOPI/STOPF commands without the PURGE option, or their AOS counterparts are handled as a non-activity timeout.

Back Out Transaction

The nucleus backs out a transaction only for an ET user that has an open transaction.

The transaction is backed out for all kinds of timeout and for STOPU commands.

Release Command IDs

When the nucleus releases command IDs, it also erases the file list and deletes the ETID in the user queue element (UQE).

This action is taken for all TNAX timeouts and for the STOPU command.

It is *not* taken for a TT timeout.

Close User Queue Element

The nucleus closes the user queue element (UQE) for all TNAX timeouts when OPENRQ=YES and for the STOPU command.

Note:

If OPENRQ=YES is in effect for the session and a user does not start with an OP command, the user receives response code 9.

This action is *not* taken for TNAX timeouts when OPENRQ=NO (with one exception described in the next paragraph) or for a TT timeout.

The nucleus closes the UQE for a TNAE timeout when OPENRQ=NO only when the user

- has no ETID;
- has no open transaction; and
- has not issued an OP command with R option.

Note:

When OPENRQ=NO and the UQE is closed, the user's next call does not receive a response code 9 because the nucleus no longer knows that it timed out the user.

Set Timeout Status in the UQE

The nucleus sets the timeout status when the UQE is not closed.

Note:

If a user issues a call and the timeout status is set in the UQE, the user receives response code 9.