Adabas

Adabas Programmer's Reference Summary

Version 8.6.2

October 2026

ADABAS & NATURAL

This document applies to Adabas Version 8.6.2 and all subsequent releases.

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

Copyright © 1971-2026 Software GmbH, Darmstadt, Germany and/or its subsidiaries and/or its affiliates and/or their licensors.

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

Detailed information on trademarks and patents owned by Software GmbH and/or its subsidiaries is located at https://softwareag.com/licenses.

Use of this software is subject to adherence to Software GmbH's licensing conditions and terms. These terms are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software GmbH Products / Copyright and Trademark Notices of Software GmbH Products". These documents are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software GmbH.

Document ID: ADAMF-PGMCARD-862-20261013

Table of Contents

Preface v
1 Conventions 1
2 About this Documentation 3
    Document Conventions 4
    Online Information and Support 4
    Data Protection 5
3 Adabas Direct Call Syntax 7
    ACB Interface Call Syntax 8
    ACBX Interface Call Syntax 8
    Specifying an ACBX Interface Direct Call in Open System Applications 9
4 Adabas Control Block Structure 11
    ACB Control Block 12
    ACBX Control Block 13
5 Adabas Buffer Description (ABD) Structure 15
6 Adabas Direct Call Buffers 17
    Format Buffers 20
    Record Buffers 31
    Multifetch Buffers 34
    Search Buffers 35
    Value Buffers 44
    ISN Buffers 46
7 Adabas Commands 47
    Adabas Command ACBs 50
    ACB Command Buffers 52
    Adabas Command ACBXs 52
    ACBX Command ABDs and Buffers 52
    A1 Command -- Update Record 52
    BT Command -- Back Out Transaction 55
    C1 Command -- Record Checkpoint 57
    C3 Command -- Write Checkpoint 59
    C5 Command -- Write User Data to PLOG 61
    CL Command -- Close Session 64
    E1 Command -- Delete Record 66
    ET Command -- End Transaction 68
    HI Command -- Hold Record 71
    L1/L4 Commands -- Read Record 73
    L2/L5 Commands -- Read Physical Sequential Record 75
    L3/L6 Commands -- Read Logical Sequential Record 78
    L9 Command -- Read Descriptor Values 81
    LF Command -- Read Field Definitions 84
    N1/N2 Commands -- Add New Record 86
    OP Command -- Open User Session 89
    RC Command -- Release Command ID or Global Format ID 92

RE Command -- Read ET User Data 94
RI Command -- Release Held Record 96
S1/S2/S4 Commands -- Find Records 98
S5 Command -- Find Coupled ISNs 101
S8 Command -- Process ISN Lists 103
S9 Command -- Sort ISN List 105
Index 109

Preface

This document summarizes the direct call syntax as well as the structure and use of Adabas control blocks (ACBs and ACBXs), Adabas buffer descriptions (ABDs), and buffers in Adabas direct calls. In addition, the structure of the control blocks and structure required for each Adabas command is given. This information is useful to the developers of the applications that use or maintain the data in your Adabas database.

The following summarizes the topics covered by this documentation:

Adabas Direct Call Syntax	Provides the syntax of Adabas ACB and ACBX direct calls.
Adabas Control Block Structure	Provides the structure of the ACB and ACBX control blocks used in ACB and ACBX direct calls.
Adabas Buffer Description (ABD) Structure	Provides the structure of the Adabas buffer descriptions (ABDs) used in ACBX direct calls.
Adabas Direct Call Buffers	Provides an overview of the buffers that can be specified in a ACB and ACBX direct calls as well as links to the syntax, requirements and use of each buffer in direct calls.
Adabas Commands	Provides overview tables of the available Adabas commands, including links to the ACB and ACBX control block structures and buffer areas of each.

1 Conventions

This document describes the syntax conventions used in this documentation for direct calls, buffer descriptions, and buffer specifications. Notation specific to a particular buffer is introduced in the discussion of that area later in this section.

Convention	Identifies...	Description	Example
uppercase, bold	an Adabas keyword	Syntax elements appearing in uppercase and bold font are Adabas keywords. When specified, these keywords must be entered exactly as shown.	`field NC` The syntax element NC is an Adabas keyword.
lowercase, italic, normal font	a variable	Syntax elements appearing in lowercase and normal, italic font identify items that you must supply.	`field i[-j]` The syntax elements *field*, *i*, and *j* are variables. They identify a value you must supply.
vertical bars (|)		Vertical bars are used to separate mutually exclusive choices. **Note:** In more complex syntax involving the use of large brackets or braces, mutually exclusive choices are stacked instead.	`a | b` In the example above, you must select between "a" or "b". There are no defaults.
brackets ([ ])	optional elements or choices	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.	`field [, length]` In this example, the length parameter is optional.
braces ({ })	required elements or choices	Braces are used to identify required elements. When multiple elements are stacked or separated by vertical bars within braces, one and only one of the elements must be supplied.	`{C | S}` In this example, either "C" or "S" must be specified.

Convention	Identifies...	Description	Example
indentation	subparameters	Indentation is used to identify subparameters of a parameter.	
ellipsis (...)	repeated elements	Ellipses are used to identify elements that can be repeated. If the term preceding the ellipsis is an expression enclosed in square brackets or braces, the ellipsis applies to the entire bracketed expression.	`[FIELD='field-name ↩ [, option]...']...` In this example, the FIELD parameter can be repeated. In addition, more than one option can be associated with a field.
b...	blank	The italicized, lowercase letter *b*, when used singly or in groups (such as *bbbbbbbb*) indicates a blank or a series of blanks. Each *b* represents one blank.	If the value "ISN*bbbbb*" is specified in this field, it indicates that the ISN values are to be used as the sorting sequence (*bbbbb* represent blanks).
other punctuation and symbols	required punctuation	All other punctuation and symbols must be entered exactly as shown.	`fmtsel redfmt.` In this example, the period is required.

2 About this Documentation

- Document Conventions 4
- Online Information and Support 4
- Data Protection 5

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
`Monospace font`	Identifies service names and locations in the format *`folder.subfolder.service`*, APIs, Java classes, methods, properties.
`Italic`	Identifies: Variables for which you must supply values specific to your own situation or environment. New terms the first time they occur in the text. References to other documentation sources.
`Monospace font`	Identifies: Text you must type in. Messages displayed by the system. Program code.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information and Support

Product Documentation

You can find the product documentation on our documentation website at **https://documentation.softwareag.com**.

Product Training

You can find helpful product training material on our Learning Portal at **https://learn.softwareag.com**.

Tech Community

You can collaborate with Software GmbH experts on our Tech Community website at **https://tech-community.softwareag.com**. From here you can, for example:

- Browse through our vast knowledge base.
- Ask questions and find answers in our discussion forums.
- Get the latest Software GmbH news and announcements.
- Explore our communities.
- Go to our public GitHub and Docker repositories at **https://github.com/softwareag** and **https://hub.docker.com/publishers/softwareag** and discover additional Software GmbH resources.

Product Support

Support for Software GmbH products is provided to licensed customers via our Empower Portal at **https://empower.softwareag.com**. Many services on this portal require that you have an account. If you do not yet have one, you can request it at **https://empower.softwareag.com/register**. Once you have an account, you can, for example:

- Download products, updates and fixes.
- Search the Knowledge Center for technical information and tips.
- Subscribe to early warnings and critical alerts.
- Open and update support incidents.
- Add product feature requests.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

3 Adabas Direct Call Syntax

- ACB Interface Call Syntax 8
- ACBX Interface Call Syntax 8
- Specifying an ACBX Interface Direct Call in Open System Applications 9

ACB Interface Call Syntax

```
CALL 'ADABAS' USING acb-control-block-name
                    [format-buffer]
                    [record-buffer]
                    [search-buffer]
                    [value-buffer]
                    [ISN-buffer]
```

Replace	With
acb-control-block-name	The pointer to the Adabas Control Block (ACB) to use for the call.
format-buffer	The name of or pointer to the format buffer to use for the call. Only one format buffer can be specified in a single ACB direct call.
ISN-buffer	The name of or pointer to the ISN buffer to use for the call. Only one ISN buffer can be specified in a single ACB direct call.
record-buffer	The name of or pointer to the record buffer to use for the call. Only one record buffer can be specified in a single ACB direct call.
search-buffer	The name of or pointer to the search buffer to use for the call. Only one search buffer can be specified in a single ACB direct call.
value-buffer	The name of or pointer to the value buffer to use for the call. Only one value buffer can be specified in a single ACB direct call.

ACBX Interface Call Syntax

```
CALL 'ADABAS' USING acbx-control-block-name
                    reserved-fullword
                    reentrancy-token
                  [format-buffer-ABD record-buffer-ABD [multifetch-buffer-ABD]]...
                    [search-buffer-ABD]
                    [value-buffer-ABD]
                    [ISN-buffer-ABD]
                    [performance-buffer-ABD]
                    [user-buffer-ABD]
```

Replace	With
acbx-control-block-name	The pointer to the extended Adabas control block (ACBX) to use for the call.
format-buffer-ABD	The name of or pointer to the format buffer ABD that defines a format buffer segment to use for the call. Each format buffer segment must end with a period and be a complete and valid standalone format buffer. Multiple format buffer ABDs can be specified in a single ACBX direct call.

Replace	With
ISN-buffer-ABD	The name of or pointer to the ISN buffer ABD that defines an v segment to use for the call. Only one ISN buffer ABD can be specified in a single ACBX direct call.
multifetch-buffer-ABD	The name of or pointer to the multifetch buffer ABD that defines a multifetch buffer segment to use for the call. Multiple multifetch buffer ABDs can be specified in a single ACBX direct call.
performance-buffer-ABD	The name of or pointer to the performance buffer ABD that defines a performance buffer segment used by Adabas Review. The performance buffer segment is reserved for use by Adabas Review.
record-buffer-ABD	The name of or pointer to the record buffer ABD that defines a record buffer segment to use for the call. Multiple record buffer ABDs can be specified in a single ACBX direct call.
reentrancy-token	The ADALNK reentrancy token. This is a fullword in the calling program's storage where ADALNK stores the address of its static data area. This fullword should be set to zero before the first Adabas call. It should then remain unchanged for all subsequent direct calls while the program runs.
reserved-fullword	The fullword containing binary zeros. This fullword is reserved for use by Adabas and should be set to binary zeros before the first Adabas call.
search-buffer-ABD	The name of or pointer to the search buffer ABD that defines a search buffer segment to use for the call. Only one search buffer ABD can be specified in a single ACBX direct call.
user-buffer-ABD	The name of or pointer to the user buffer ABD that defines a user buffer segment (extension) to use for the call. The user buffer extension (UBX) is used for the user data passed to user exits LNKUEX1 (link routine pre-call exit) and LNKUEX2 (link routine post-call exit). A single user buffer ABD can be specified in an ACBX direct call.
value-buffer-ABD	The name of or pointer to the value buffer ABD that defines a value buffer segment to use for the call. Only one value buffer ABD can be specified in a single ACBX direct call.

Specifying an ACBX Interface Direct Call in Open System Applications

```
CALL 'ADABAS' USING acbx-control-block-name
                    ABD-count
                    ABD-list-pointer
```

Replace	With	Conditions
`acbx-control-block-name`	The pointer to the extended Adabas control block (ACBX) to use for the call.	Required.
`ABD-count`	The number of ABD pointers included in the ABD list for the direct call.	Required only if ABDs and their associated buffers are used in the direct call.
`ABD-list-pointer`	The pointer to the ABD list for the direct call. The ABD list contains pointer references for all of the ABDs used by the ACBX direct call. For more information about the ABD list, read *ABD Lists*, elsewhere in this guide.	Required only if buffers are required for the direct call.

4 Adabas Control Block Structure

- ACB Control Block 12
- ACBX Control Block 13

ACB Control Block

DSECT Name	Field	Control Block Position	Offset	Length (in Bytes)	Format
ACBTYPE	Call Type	1	00	1	binary
reserved	(reserved)	2	01	1	binary
ACBCMD	Command Code	3-4	02	2	alphanumeric
ACBCID	Command ID	5-8	04	4	alphanumeric / binary
ACBFNR	File Number	9-10	08	2	binary
ACBRSP	Response Code	11-12	0A	2	binary
ACBISN	ISN	13-16	0C	4	binary
ACBISL	ISN Lower Limit	17-20	10	4	binary
ACBISQ	ISN Quantity	21-24	14	4	binary
ACBFBL	Format Buffer Length	25-26	18	2	binary
ACBRBL	Record Buffer Length	27-28	1A	2	binary
ACBSBL	Search Buffer Length	29-30	1C	2	binary
ACBVBL	Value Buffer Length	31-32	1E	2	binary
ACBIBL	ISN Buffer Length	33-34	20	2	binary
ACBCOP1	Command Option 1	35	22	1	alphanumeric
ACBCOP2	Command Option 2	36	23	1	alphanumeric
ACBADD1	Additions 1	37-44	24	8	alphanumeric / binary
ACBADD2	Additions 2	45-48	2C	4	alphanumeric / binary
ACBADD3	Additions 3	49-56	30	8	alphanumeric
ACBADD4	Additions 4	57-64	38	8	alphanumeric
ACBADD5	Additions 5	65-72	40	8	alphanumeric / binary
ACBCMDT	Command Time	73-76	48	4	binary
ACBUSER	User Area	77-80	4C	4	not applicable

ACBX Control Block

DSECT Field Name	Field	Control Block Position	Offset	Length (in bytes)	Format
ACBXTYP	Call Type	1	00	1	binary
ACBXRSV1	Reserved 1	2	01	1	binary
ACBXVER	Version Indicator	3-4	02	2	binary
ACBXLEN	ACBX Length	5-6	04	2	binary
ACBXCMD	Command Code	7-8	06	2	alphanumeric
ACBXRSV2	Reserved 2	9-10	08	2	binary
ACBXRSP	Response Code	11-12	0A	2	binary
ACBXCID	Command ID	13-16	0C	4	alphanumeric/ binary
ACBXDBID	Database ID	17-20	10	4	numeric
ACBXFNR	File Number	21-24	14	4	numeric
ACBXISNG	8-Byte ISN	25-32	18	8	do not use
ACBXISN	ISN	29-32	1C	4	binary
ACBXISLG	8-Byte ISN Lower Limit	33-40	20	8	do not use
ACBXISL	ISN Lower Limit	37-40	24	4	binary
ACBXISQG	8-Byte ISN Quantity	41-48	28	8	do not use
ACBXISQ	ISN Quantity	45-48	2C	4	binary
ACBXCOP1	Command Option 1	49	30	1	alphanumeric
ACBXCOP2	Command Option 2	50	31	1	alphanumeric
ACBXCOP3	Command Option 3	51	32	1	alphanumeric
ACBXCOP4	Command Option 4	52	33	1	alphanumeric
ACBXCOP5	Command Option 5	53	34	1	alphanumeric
ACBXCOP6	Command Option 6	54	35	1	alphanumeric
ACBXCOP7	Command Option 7	55	36	1	alphanumeric
ACBXCOP8	Command Option 8	56	37	1	alphanumeric
ACBXADD1	Additions 1	57-64	38	8	alphanumeric/ binary
ACBXADD2	Additions 2	65-68	40	4	binary
ACBXADD3	Additions 3	69-76	44	8	alphanumeric/ binary
ACBXADD4	Additions 4	77-84	4C	8	alphanumeric
ACBXADD5	Additions 5	85-92	54	8	alphanumeric/ binary
ACBXADD6	Additions 6	93-100	5C	8	alphanumeric/ binary
ACBXRSV3	Reserved 3	101-104	64	4	binary
ACBXERRG	Error Offset in Buffer (64-bit)	105-112	68	8	do not use

DSECT Field Name	Field	Control Block Position	Offset	Length (in bytes)	Format
ACBXERRA	Error Offset in Buffer (32-bit)	109-112	6C	4	binary
ACBXERRB	Error Character Field	113-114	70	2	alphanumeric
ACBXERRC	Error Subcode	115-116	72	2	binary
ACBXERRD	Error Buffer ID	117	74	1	alphanumeric
ACBXERRE	Reserved for future use	118	75	1	do not use
ACBXERRF	Error Buffer Sequence Number	119-120	76	2	numeric
ACBXSUBR	Subcomponent Response Code	121-122	78	2	binary
ACBXSUBS	Subcomponent Response Subcode	123-124	7A	2	binary
ACBXSUBT	Subcomponent Error Text	125-128	7C	4	alphanumeric
ACBXLCMP	Compressed Record Length	129-136	80	8	binary
ACBXLDEC	Decompressed Record Length	137-144	88	8	binary
ACBXCMDT	Command Time	145-152	90	8	binary
ACBXUSER	User Area	153-168	98	16	not applicable
ACBXRSV4	Reserved 4	169-193	A8	24	*do not touch*

5 Adabas Buffer Description (ABD) Structure

DSECT Field Name	Field	Control Block Position	Offset	Length (in bytes)	Format
ABDXLEN	ABD length	1-2	00	2	binary
ABDXVER	Version indicator	3-4	02	2	alphanumeric
ABDXID	Buffer Type ID	5	04	1	alphanumeric
ABDXRSV1	Reserved 1	6	05	1	binary
ABDXLOC	Buffer location flag	7	06	1	alphanumeric. although a binary zero (x'00') is tolerated instead of a blank.
ABDXRSV2	Reserved 2	8	07	1	binary
ABDXRSV3	Reserved 3	9	08	4	binary
ABDXALET	ALET for buffer (if ABDXLOC=C'D')	13	0C	4	binary
ABDXSIZE	Buffer size (allocated length)	17-24	10	8	binary
ABDXSEND	Data length to send from client to the nucleus	25-32	18	8	binary
ABDXRECV	Data length received by the client from the nucleus	33-40	20	8	binary
ABDXADRG	64-bit indirect address pointer (if ABDXLOC=C'I' or C'D')	41-48	28	8	binary
ABDXADR	34-bit indirect address pointer (if ABDXLOC=C'I' or C'D')	45-48	2C	4	binary
---	Buffer (if ABDXLOC=C' ' or X'00')	49-n	30	user-defined	not applicable

6 Adabas Direct Call Buffers

- Format Buffers 20
- Record Buffers 31
- Multifetch Buffers 34
- Search Buffers 35
- Value Buffers 44
- ISN Buffers 46

The following syntax depicts the relationships between the different types of buffers that can be specified for a direct call. It should assist you in determining which buffer specifications are dependent on the presence of others.

```
[format-buffer record-buffer... [multifetch-buffer]]...
[search-buffer value-buffer]
[ISN-buffer]
[user-buffer] ...
[performance-buffer]
```

Notes:

1. If you are specifying an ACBX interface direct call, corresponding Adabas buffer descriptions (ABDs) must also be specified. In addition, in ACBX interface direct calls when buffer specifications require the presence of other buffer specifications (for example, a format buffer requires the presence of a record buffer), Adabas pairs the buffers in the sequence in which they are specified (for example, the first specified format buffer ABD with the first specified record buffer ABD). The syntax below can assist you in determining the sequence in which the ABDs should be listed in the call or in the ABD list.
2. If you are specifying an ACB interface direct call, the multifetch, performance, and user buffers listed in this syntax do not apply. In addition, buffers must be specified in this sequence: format, record, search, value, and ISN. If an earlier buffer in the sequence is not needed, but a later one is, all of the buffers up to the needed buffer must be specified, even if they are blank. For example, if an ACB interface direct call requires an ISN buffer but none of the other buffers, dummy format, record, search, and value buffers must be specified before the ISN buffer.

The following table describes the elements in this syntax:

Element	Description	Conditions
format-buffer	A format buffer segment to use for the call. Each format buffer segment must end with a period and be a complete and valid standalone format buffer.	Required only if you need to specify the fields to be processed during the execution of an Adabas read or update command. When required, multiple format buffers can be specified for an ACBX interface direct call. Only one format buffer can be specified in an ACB interface direct call. If a format buffer is specified in the call, a corresponding record buffer must also be specified. In an ACBX interface direct call, if a record buffer is not provided, Adabas will create a dummy one (with length zero) to pair with the format buffer. In an ACB interface direct call, if a record buffer is not provided, processing errors will occur.

Element	Description	Conditions
		Optionally, in an ACBX interface direct call, a corresponding multifetch buffer can also be specified.
`ISN-buffer`	An ISN buffer segment to use for the call.	Required only if you need to set aside an area in storage to store ISNs or (in the case of an ACB interface direct call) an area to store the record descriptor elements (RDEs) of multifetched or prefetched records. When required, only one ISN buffer should be specified for the call.
`multifetch-buffer`	A multifetch buffer segment to use for the ACBX interface direct call. This buffer is only available for ACBX interface direct calls.	Used only by ACBX interface direct calls and required only if you need to set aside an area in storage to store the record descriptor elements (RDEs) of multifetched records. When required, multiple multifetch buffers can be specified for an ACBX interface direct call. If a multifetch buffer is specified, corresponding format and record buffers must also be specified. If they are not, Adabas will create dummy format and record buffers (with length zero) to correspond with the multifetch buffer.
`performance-buffer`	A performance buffer to use for the ACBX interface direct call. This buffer is only available for ACBX interface direct calls and is only used by Adabas Review.	Not required. Used only by ACBX interface direct calls withAdabas Review. For more information, read the Adabas Review documentation.
`record-buffer`	A record buffer segment to use for the call.	Required only if you need to set aside an area of storage to store record data required or collected for the call. When required, multiple record buffers can be specified for an ACBX interface direct call. Only one record buffer can be specified in an ACB interface direct call. If a record buffer is specified in the call, a corresponding format buffer must also be specified. In an ACBX interface direct call, if a format buffer is not provided, Adabas will create a dummy one (with length zero) to pair with the record buffer. In an ACB interface direct call, if a format buffer is not provided, processing errors will occur. Optionally, in an ACBX interface direct call, a corresponding multifetch buffer can also be specified.

Element	Description	Conditions
search-buffer	A search buffer segment to use for the call.	Required only if search criteria are required to select records for the call. If a search buffer is specified in the call, a corresponding value buffer must also be specified. Only one search and value buffer pair can be specified in a single direct call.
user-buffer	A user buffer segment (extension) to use for the call. The user buffer extension (UBX) is used for the user data passed to Adabas nucleus user exits 11 and 4 and Adalink user exits 1 and 2 (user exits A and B in Adabas 7).	Used only by ACBX interface direct calls and required only if the call requires input for the Adabas nucleus user exits 11 and 4 and the Adalink user exits 1 and 2 (user exits A and B in Adabas 7). You can specify a single user buffer in a direct call.
value-buffer	A value buffer segment to use for the call.	Required only if search criteria are required to select records for the call. If a value buffer is specified in the call, a corresponding search buffer must also be specified. Only one search and value buffer pair can be specified in a single direct call.

Format Buffers

The format buffer has the following syntax:

```
[field-selection-criteria1] record-format1[,[field-selection-criteria2] record-format2]... ↩
.
```

A comma must be used to separate all format buffer entries. One or more spaces may be present between entries. The last entry may not be followed by a comma.

The format buffer must end with a period.

- Field Selection Criteria
- Record Format Specifications
- field Syntax
- Length and Data Format
- Field Series Notation
- Space Notation (nX)

- Text Insertion Notation

Field Selection Criteria

Field selection criteria (*field-selection-criteria1* and *field-selection-criteria2*) are optional in a format buffer. They allow you to restrict record formats to specific values of fields. The syntax of field selection criteria is:

```
(field-name operator value1 [, value2]...)
```

field-name

The field name used in field selection criteria must be the valid name of a field in the FDT of the Adabas file being read. It *cannot* be:

- The name of a group or periodic group
- A field using any of the MU, PE, LA, or LB options
- A subfield, superfield, subdescriptor, or superdescriptor
- A collation descriptor
- A hyperdescriptor
- A phonetic descriptor.

In addition, fields specified with the NU or NC/NN options must have a non-null value; otherwise the selection criteria will be false.

operator

The following table lists the operators that can be used in the format buffer field selection criteria.

Operator	Meaning
EQ	equal to
=	equal to
NE	not equal to
LT	less than
<	less than
GT	greater than
>	greater than
LE	less than or equal to
GE	greater than or equal to

value1, value2

The value must be a numeric integer or an alphanumeric value.

If you use the EQ or = operator, a series of values can be specified, separated by commas. An alphanumeric value must be enclosed within apostrophes (for example, `'value'`).

Consider the following example:

```
(SA = 1) record-format-1, (SA = 2,3,4) record-format-2, (SA GE 4) record-format-3.
```

The field selection criteria specifies that:

- If the value of field SA is "1", record-format-1 is used;
- If the value of field SA is "2", "3" or "4", record-format-2 is used;
- If the value of field SA is equal to or greater than "4", record-format-3 is used.

The first criterion that is met is used. If no criteria are met, a response code is returned. In the example above, if the value of field SA is "4", both the last two conditions in the field selection criteria are satisfied. However, record-format-2 will always be used, rather than record-format 3 because it is the first criteria satisfied. Likewise, if the value of SA is "0", a response code is returned.

Record Format Specifications

The record format (*`record-format1`*) is required and is used to indicate which fields should be read or updated in the Adabas direct call. Additional record formats can also be specified (*`record-format2`*).

The syntax of the record format is:

```
{ field [, length ] [, data-format ]
  field-name - field-name
  nX
  'text'                            }
```

For information about each of the elements in this syntax, read the section listed in the following table:

Syntax Element	Read
`field`	*`field` Syntax*
`length`	*Length and Data Format*
`data-format`	*Length and Data Format*
`field-name` - *`field-name`*	*Field Series Notation*
*n*X	*Space Notation (nX)*
'*`text`*'	*Text Insertion Notation*

field Syntax

The syntax for a `field` in record format syntax is:

```
                 [ C                                                  ]
                 [ D [ mu-pe-indices ]                                ]
                 [ L                                                  ]
                 [ 1- n                                               ]
                 [ S                                                  ]
field-name       [ i   [ C                                            ]
                 [     [ [ - j ] [ ( { [1-] N  } ) ]                  ]
                 [     [           { i [ - j ] }                      ]
                 [                                                    ]
                 [ N   [ C                                            ]
                 [     [ (     { [1-] N  } )                          ]
                 [     [         { i [ - j ] }                        ]
```

where:

`field-name` **Specifications**

The `field-name` is the name of the field or group for which the value, range, or count is requested or for which a new value is being provided. The name specified must be two characters in length and must be present in the FDT of the file being read or updated by the Adabas direct call. The name can refer to an elementary, subfield, superfield, multiple-value field, a group, or a periodic group.

A field name that refers to a group results in all the fields within the group being referenced. Use of group names can greatly reduce the time required to process the command. A group name cannot be used if the group contains a multiple-value or variable-length field (no standard length).

For access commands, the same name may be specified more than once. In this case, the field value is returned multiple times.

For update commands, the same name cannot be used more than once (except in the case of multiple-value fields, as explained later in this section).

A subfield, superfield, subdescriptor, or superdescriptor name may be specified for access commands but not for update commands.

Index or Range Notation (*i* [-*j*] Notation)

The following is the field name syntax for selecting multiple-value fields or occurrences of periodic groups:

```
field-name i[-j]
```

where:

i	is the periodic group or multiple-value occurrence
i-j	is the periodic group or multiple-value occurrence range

Periodic group names *must* be followed by a numeric or other appropriate suffix (see the discussions of the *Count Indicator (C)* and the highest occurrence/value indicator *Highest Occurrence/Value Indicator (N)* for more information). Specifying a periodic group name as the field name alone is incorrect syntax.

Multiple-value fields can be specified by explicitly identifying a particular value (indexing) or by referencing each value in sequence, letting Adabas assign an index based on the sequence.

Count Indicator (C)

To obtain the count of periodic group occurrences, or the count of existing values of a multiple-value field not in a periodic group, specify the periodic group or multiple-value field name followed by "C":

```
field-name C ↩
```

Daylight Savings Indicator (D)

If a user session time zone uses daylight savings time, the user's local time requires a daylight savings indicator to ensure that the accurate time is reported during the hour when time is turned back from daylight savings time to standard time. Specifying the daylight savings indicator (D) on a date-time field can be used to provide you with information about whether the field's date-time value is in standard or daylight savings time.

The syntax of the daylight savings indicator is simply the letter "D" specified after the field name and before any optional MU index, PE index, or MU-PE index combinations:

```
fnD[mu-pe-index]
```

The daylight savings indicator can only be specified for date-time fields defined with the TZ option. When a daylight savings indicator is specified for a field in a format buffer, it must have a format of 2,F. The offset from standard time (in seconds) is specified in the record buffer associated with the format buffer in the following format:

```
H'nnnn'
```

The value *nnnn* represents the number of seconds that the stored time should be offset from standard time to calculate daylight savings time. A value of zero indicates that standard time should be used; any value other than zero indicates that daylight savings time should be adjusted for and specifies the offset for that adjustment.

It is possible that the daylight savings time offset is negative if the previous standard time belongs to a different time zone and the regrouping coincided with the change to daylight savings time. In addition, any security-by-value criteria on a date-time field defined with the TZ option will be evaluated as given in UTC time.

Length Indicator (L)

The format buffer indicator, *L*, can be used to retrieve or specify the actual length of any LA or LB alphanumeric or wide-character field value. This format buffer element is referred to as the *length indicator*.

Note: At this time, the length indicator can only be used in format buffer specifications for LA or LB fields. Support for use of the length indicator in other fields as well will be considered in a future release of Adabas.

Highest Occurrence/Value Indicator (N)

The indicator "N" selects the last value in a series of values comprising a multiple-value field, or the last occurrence of a periodic group, removing the need to know the number of the last value or occurrence.

Note: The `1-N` notation is not supported for LB fields.

The notation `1-N` selects all values comprising a multiple-value field, or all occurrences of a periodic group. For multiple-value fields in periodic groups, it is not possible to combine the specification 1-N for the group occurrence with any specification for the field occurrences.

The notation `NC` selects the count of the existing values of a multiple-value field in the last occurrence of the periodic group containing the field.

SQL Significance Indicator (S)

The S significance, or null, indicator and the corresponding null indicator value in the record buffer indicate whether a field's value is significant, including zero or blank, or not significant (undefined). The S indicator can only be applied to elementary fields that are defined with the NC option, but not for an NU option field:

```
field-name S
```

Length and Data Format

The length and format parameters are used if a field value is being provided or is to be returned in a length or format different from the standard defined for the field in the FDT. If the length or format parameters are omitted, the field value must be provided or is returned in the standard length and format of the field:

```
[ , length ] [ , data-format ]
```

Possible format and length conversions are suggested by the information in the following table. A format conversion cannot be specified for subfields or subdescriptors; superfields or superdescriptors; or hyperdescriptors.

Fmt	Max Length (in bytes)	Data Type	Compatible Formats
A	253	alphanumeric, left-justified	W
W	253 [1]	wide-character, left-justified	A
A,W	16,381 [1, 2, 3]	alphanumeric or wide-character with LA (long alpha) option; left-justified; preceded by optional two-byte binary (inclusive) length	W,A
A	2,147,483,643[3, 4]	alphanumeric with LB (LOB) option; left-justified; preceded by optional four-byte binary (inclusive) length	None
B	126	binary; right-justified; unsigned	A,F,P,U
F	8	fixed-point; right-justified; signed; two, four, or eight bytes	A,B,P,U
G	8	floating-point; four or eight bytes	none
P	15	packed decimal; signed; positive=A,C,E, or F; negative=B or D	A,B,F,U
U	29	unpacked decimal; signed; positive=A,C,E, or F; negative=B or D	A,B,F,P

Note:

1. Like an alphanumeric field, a wide-character field may be a standard length in bytes defined in the FDT, or variable length. Any non-variable length override for a wide-character field must be compatible with the user encoding; for example, a user encoding in Unicode requires an even length (maximum of 252 bytes for non-LA fields, maximum of 16,380 bytes for LA fields).
2. If the LA value in the record buffer is preceded by its two-byte length (variable field length notation), the maximum length value is 16,383 bytes.
3. For LA and LB fields only, you can specify an asterisk (*) instead of a length in the format element. For more information, read ***Asterisk (*) Length Notation***, elsewhere in this section.
4. If the LOB value in the record buffer is preceded by its four-byte length (variable field length notation), the maximum length value is 2,147,483,647 bytes.

The length specified must be large enough to contain the value in the chosen format, but cannot exceed the maximum length permitted.

If a length of zero is specified, or if no length is specified and `field-name` refers to a variable-length field (no standard length), the amount of record buffer space used for the field is variable and depends on its actual value. The value returned by Adabas begins with a length indicator that specifies the value's length in binary format, including the space for the indicator itself. For update commands, you must provide this length indicator at the beginning of the field value in the record buffer.

- For a field without the LA- and LB-options, the length indicator is one byte.
- For an LA field, the length indicator is two bytes.

- For an LB field, the length indicator is four bytes.

For example, if the length in the format buffer is given as zero, an LA field with a 1000-byte value is represented in the record buffer as two bytes containing the number 1002 in binary format, followed by 1000 bytes containing the field value proper.

The format specified must be compatible with the standard format of the field.

- Conversion between packed/unpacked decimal values and binary is limited to values between 0 and 2,147,483,647.
- Conversion from a numeric format to alphanumeric results in an unpacked value, left justified, without leading zeros and with trailing blanks. For example, the three-byte packed value "10043F" would be converted to "F1F0F0F4F3404040". Value truncation is possible with this type of conversion.

Note: Length and format overrides are not allowed in format elements for LOB segments (when LOB segment notation is used in the format buffer).

The following additional topics are covered in this section:

- Asterisk (*) Length Notation
- Edit Mask Notation (Read Operations Only)

Asterisk (*) Length Notation

For LA and LB fields only, you can specify an asterisk (*) instead of a length in the format element. Asterisks cannot be specified for regular alphanumeric or wide-character fields. The presence of an asterisk indicates that the amount of space available for the LB field value in the record buffer is variable and depends on the actual value of the LB field. However, unlike the zero length specification setting, *no* four-byte length field precedes the LB field value in the record buffer; the record buffer area corresponding to the LB format element only contains the value of the LB field. The actual LB field value length *should be* retrieved for read commands and *must be* specified for update commands using the new format buffer length indicator, *L*. For more information about the length indicator, read ***Length Indicator (L)***, elsewhere in this guide.

In the following example, the record buffer for LB field L1 contains only the value of the L1 field, followed by the value of the AA field for which 10 bytes have been allotted.

```
FB='L1,*,AA,10,A.'
```

In the following example, the record buffer for LB multi-value field L2 contains the first ten values of L2.

```
FB='L21-10,*.'
```

The record buffer is not necessarily required to provide sufficient space for the entire field if its format element includes an asterisk length setting. However, in read command processing, the field value can be truncated if *both* of the following conditions are met:

- The record buffer space available is insufficient for the field value.
- A field with asterisk notation is specified at the *end* of the format buffer.

In these conditions, no error is returned. If this were the case in the second example above (`FB='L21-10,*.'`), Adabas would truncate the ten values to be read down to the length of the corresponding record buffer segment. (The truncation occurs from right to left; that is, the last value is truncated first; if the remaining space is still insufficient, the second-to-last value is truncated, and so on.) In extreme cases, if no space is available at all for the field value, the value is truncated down to zero bytes.

In the first example above (`FB='L1,*,AA,10,A.'`), if the record buffer segment is too short, no truncation occurs because this is not allowed for fields specified with a fixed length or length of zero (0). Rather, the nucleus returns response code 53 (ADARSP053), indicating that the record buffer is too small.

Only read commands executed by the Adabas nucleus may truncate values specified with the asterisk notation; no truncation occurs in update commands. In addition, the ADACMP utility does not truncate values specified with the asterisk notation.

Edit Mask Notation (Read Operations Only)

Edit masks are used according to the standard edit mask rules used in the COBOL programming language.

An edit mask may only be specified for numeric fields. All data returned by Adabas to an edited field is converted to unpacked decimal format regardless of the standard format of the field. A maximum of 15 digits (not counting edit characters) can be returned to an edited field.

For a field with an edit format specified, the length parameter must be large enough to contain the field value plus all required edit characters.

Format	Generates the edit mask . . .
E1	zzzzzzzzzzzzzzz
E2	zzzzzzzzzzzzz9-
E3	zzzzzzzzz99.99.99
E4	zzzzzzzzz99/99/99
E5	z.zzz.zzz.zzz.zzz,zz
E6	z,zzz,zzz,zzz,zzz.zz

Format	Generates the edit mask . . .
E7	z,zzz,zzz,zzz,zz9.99-
E8	z.zzz.zzz.zzz.zz9,99-
E9	*,***,***,***,**9.99-
E10	*.***.***.***.**9,99-
E11	user-designated mask
E12	user-designated mask
E13	user-designated mask
E14	user-designated mask
E15	user-designated mask

Note: Although edit formats E3 and E4 provide space for the century digits (see the following examples), they do not *enforce* date formats that are compatible with year 2000 requirements.

For information on date-time edit masks, supplied with Adabas, read *Date-Time Edit Mask Reference*, in the *Adabas DBA Tasks Manual*. Date-time edit masks can be used for record updates, in addition to reads. For information on how these date-time edit masks can be used and processed in format and search buffers, read *Date-Time Edit Mask Processing in Format and Search Buffers*, elsewhere in this guide.

Field Series Notation

The notation `field-name - field-name` may be used to refer to a series of consecutive fields (as ordered in an FDT). The user specifies the beginning and ending field names connected by a dash:

```
field-name - field-name
```

No multiple-value field or periodic group may be contained within the series.

A name that refers to a group may not be specified as the beginning or ending name, but a group may be embedded within the series.

Standard format and length is in effect for all the fields within the series. No length or format override is permitted.

The SQL Significance Indicator and Field Series Notation

When a group or range of fields contains a field specified with the NC option, the corresponding S operator is optional for read (L*x*) commands. For update (A1) commands, the S operator must not be specified. Adabas assumes that the null indicator corresponding to the NC field in the format buffer is located just in front of the field's value in the record buffer.

For example, given the following field definitions in the FDT:

```
01,GR
  02,AA,8,A
  02,BB,8,A,NC
  02,CC,8,A
```

if the format buffer of an update-type command specifies GR. or AA-CC. , the record buffer has the following structure:

```
AA-value null-indicator-BB BB-value CC-value
```

That is, the null indicator must be included in the record buffer sequence, although the S indicator was not (and must not be) specified in the format buffer.

If the format buffer of a read (L*x*) command specifies GR,BBS. or AA-CC,BBS., the record buffer has the following structure:

```
AA-value null-indicator-BB BB-value CC-value
null-indicator-BB
```

In other words, the first appearance of the null indicator is implied in the record buffer while the second appearance was explicitly called for by the format buffer.

Space Notation (nX)

The nX syntax is used differently for read and update commands:

```
nX
```

For read commands, *n*X indicates that *n* spaces are to be inserted in the record buffer by Adabas immediately before the next field value:

For update commands, *n*X causes *n* positions in the record buffer to be ignored by Adabas:

Text Insertion Notation

The text syntax is used differently for read and update commands:

```
'text'
```

For read commands, the character string specified in the format buffer is to be inserted in the record buffer immediately before the next field value. The character string provided can be 1-255 bytes long, and may contain any alphanumeric character except a quotation mark.

For update commands, the number of positions enclosed within the apostrophes in the format buffer will be ignored in the corresponding positions of the record buffer.

Record Buffers

A record buffer defines an area in storage to which Adabas can return data or in which you supply data for processing. When a record buffer is required, a corresponding format buffer is expected as well. If a format buffer is not provided, Adabas will create a dummy format buffer (with length zero) to pair with the record buffer.

When using the ACBX direct call interface, multiple record buffers can be specified for an Adabas direct call.

Record buffers are used primarily with read, search, and update commands:

- For read commands, the values of the fields specified in the format buffer are returned by Adabas in the record buffer. They are returned in the order specified by the format buffer.

  Each value is returned in the standard length and format defined for the field unless a length or format override was specified in the format buffer. If the value is a null value, it is returned in the format that is in effect for the field, as follows:

Field Type	Null value represented by . . .
Alphanumeric (A)	blanks (hex '40') or blank of user override encoding
Binary (B)	binary zeros (hex '00')
Fixed (F)	binary zeros (hex '00')
Floating Point (G)	binary zeros (hex '00')
Packed (P)	decimal packed zeros with sign (hex '00' followed by '0A', '0B', '0C', '0D' or '0F' in the rightmost, low-order byte)
Unpacked (U)	decimal unpacked zeros with sign (hex 'F0' followed by 'C0' or 'D0' in the rightmost, low-order byte)
Wide-character (W)	Unicode blanks (hex '20') or blank of user override encoding

  Note: SQL-compatible null values in NC/NN option fields require the additional null value and significance indicator. See *Specifying and Reading the SQL Null Indicator in Record Buffers*, and *SQL Significance Indicator (S)*.

  Adabas returns the number of bytes equal to the combined lengths (standard or overridden) of all requested fields.

- For add or update commands, the new values for the fields specified in the format buffer are provided by the user in the record buffer.

  When updating a record, you must specify the new value in the record buffer. If a null value is being provided, it must be provided according to the field type in effect, as described above.

- The record buffer is also used to transfer information between the user program and Adabas in the following commands:

Command	Data Provided	Data Returned
OP	Files to update and the operation type (ET, exclusive control)	User data (optional)
LF	-	Field definitions for the file
RE	-	User data stored in system file
C5	Protection log user data	-
ET/CL	User data (optional)	-

  For the OP command, the record buffer indicates the type of user and the files to be used.

  The record buffer is also used for user data (OP, RE, CL, ET commands).

This section covers the following topics:

- Specifying and Reading the SQL Null Indicator in Record Buffers
- Specifying Field Lengths of LA (Long Alpha) Fields in Record Buffers

Specifying and Reading the SQL Null Indicator in Record Buffers

To support Adabas SQL Gateway (ACE) and other structured query languages (SQLs), fields defined with the NC/NN (not-counted/null-not-allowed) options indicate an SQL-significant null with a two-byte binary null indicator in the record buffer.

Whether a field's zero value is significant or an irrelevant null (unspecified) depends on the null indicator specified in the record buffer when the value is entered or changed, or returned in the record buffer when the value is read.

In addition to specifying or reading the value itself, either:

- set the null indicator into the record buffer position that corresponds to the field's designation in the format buffer for an update operation, or
- ensure that your program examines the null indicator (if any) returned in the record buffer position corresponding to the field's position in the format buffer for a read operation.

The null indicator is always two bytes long and has fixed-point format, regardless of the data format.

For a read (L*x*) or find with read (S*x* with format buffer entry) command, the null indicator value returns one of the following (hexadecimal) null indicator values, according to the actual value that the selected field contains:

Value	Description
FFFF	A null value in this field is not significant.
0000	A null value in this field is a significant value; that is, a true zero or blank.
xxxx	The field was truncated. The null indicator contains the length (*xxxx*) of the entire value as stored in the database record.

For an update (Ax) or add (Nx) command, the (hexadecimal) null indicator value in the record buffer must be set to one of the following values:

Value	Description
FFFF	The field value is set to "undefined", an insignificant null; the field's contents in the record buffer are irrelevant when set to binary zero or blank characters.
0000	If either no value is specified in the record buffer, or binary zero or blanks are specified, the field contains a significant null value.

For an *add* command, if no value for the field is supplied in the record buffer for a field defined with the NC option, the field is treated as a null field. The following example shows how a null would be represented in a two-byte Adabas binary field AA defined with the NC option:

Field definition: 01,AA,2,B,NC

	For a nonzero value	For a blank	For null
Null Value indicator in Record Buffer	0 (binary value is significant)	0 (binary null is significant)	FFFF (binary null is not significant)
Data	0005	0000 (zero)	not relevant
Adabas internal representation	0205	0200	C1

For an *update* (A1/N1) command, the field value is always significant whenever the field is defined with the NC option; the field is treated as if a hexadecimal null indicator value of "0000" has been specified.

For a *read* command, if the null indicator is not specified for an NC option field, the field value is returned in the record buffer whenever there is a significant value in the record. If the Data Storage record contains a "not significant" (FFFF) indicator value for the field, response code 55 (ADARSP055) will be returned when the record is read.

Specifying Field Lengths of LA (Long Alpha) Fields in Record Buffers

The LA option is normally used with variable-length data. The length of an alphanumeric field with the LA option can be specified in the record buffer. The field value is preceded by a two-byte length field containing the length of the value, plus 2 (inclusive length).

Multifetch Buffers

Multifetch buffers are needed *only* for some Adabas commands run using the ACBX direct call interface; they are not needed for any ACB interface direct calls.

A multifetch buffer defines an area in storage to which Adabas can return the record descriptor elements (RDEs) of multifetched records. This buffer is only required by Adabas commands for which the multifetch option has been activated (by setting Command Option 1 to "M"). RDEs are each 16 bytes long.

A record descriptor element (RDE) has the structure shown in the following table.

Format	Length	Content
All fields unsigned integer, right aligned	4 bytes	Length of this record in record buffer. Records may have different lengths.
	4 bytes	Adabas response for this record. If a nonzero response is given, no record is stored in the record buffer.
	4 bytes	ISN for this record.
	4 bytes	(L9 only) ISN quantity: value count for this descriptor.

When the multifetch option *M* is set in the Command Option 1 field of an ACBX command, Adabas returns all records being read in the specified record buffer segments, based on the format specifications in the corresponding format buffer segments. For each record buffer segment, the corresponding multifetch buffer segment contains multifetch headers describing the records in the record buffer segment.

For BT or ET commands, a multifetch buffer is *not* needed if Command Option 1 is set to "M". In this case, the ISN buffer is used to store the ISNs that need to be removed from the hold queue.

When a multifetch buffer is required, a corresponding format and record buffer are expected as well. If they are not provided, Adabas will create dummy format and record buffers (with length zero) to pair with the multifetch buffer. For complete information about the relationships between the different types of ABD or buffer specifications, read *Understanding the Different Buffer Types*, in the *Adabas Command Reference Guide*.

Multiple multifetch buffers can be specified for an Adabas direct call. For complete information about multifetch processing, read *Multifetch Operation Processing*, elsewhere in this guide.

Search Buffers

Delimiters (commas, slashes, parentheses, semicolons) must separate all search buffer entries as indicated. One or more spaces may be present between entries. The search statement must end with a period.

This section covers the following topics:

- Search Expression
- Connecting Search Expressions
- Searching One File
- Searching Multiple, Physically Coupled Files
- Searching One or More Files Using Soft Coupling
- Physically Coupled Files
- Soft Coupling

Search Expression

The search expression syntax is common to all types of searches:

```
{ [ field-nameD [ i ], ] field-name [ i | S ] [ , length ] [ , format ] [ ,{ value-operator | EQ }] } [ ,connecting-operator
{ ( command-id )                                                                                  }

    { ,[ field-nameD [ i ], ] field-name [ i | S ] [ , length ] [ , format ] [ ,{ value-operator | EQ }] } ]...
    { ,( command-id )                                                                                  }
```

Each of the elements in this syntax is now described:

`field-name`

The search expression can name a field (descriptor or nondescriptor), subdescriptor, superdescriptor, hyperdescriptor, collation descriptor, or phonetic descriptor. When using nondescriptors, multiple-value fields are permitted, but sub-/superfields are not.

If a nondescriptor is used, Adabas reads the entire file in order to determine which records satisfy the search criteria. If only descriptors are used, the inverted lists are used and no reading of records is necessary. Search criteria containing nondescriptors and descriptors may be combined.

If a descriptor field is not initialized and logically falls past the end of the physical record, the inverted list entry for that record is not generated for performance reasons and therefore, the record will not be returned in a search. To generate the inverted list entry in this case, it is ne-

cessary to unload short, decompress, and reload the file; or use an application program to initialize the field for each record of the file.

If the descriptor is defined with the NU option (null-value suppression), null values are not stored in the inverted lists; therefore, a search for all the records which have the null value will always result in no records found (even if there are records in Data Storage which contain a null value for the descriptor). This rule also applies to subdescriptors. A superdescriptor value is not stored if any field from which it is derived is defined with the NU option *and* the value of that field is actually null.

Note: Large object (LB) fields cannot be specified in a search buffer, nor can they be specified in format selection criteria.

command-id
The command ID value is enclosed in parentheses and identifies a list of ISNs resulting from a previous S*x* command that specified the save-ISN-list option.

i **(Occurrence Index)**
The occurrence index (*i*) identifies a particular occurrence of a descriptor or nondescriptor within a periodic group and is used to limit the search to only the values located in the specified occurrence. If no index is provided, the values in all occurrences are searched.

- The index comprises one to five digits; leading zeros are permitted.
- An index is *not* permitted for a descriptor that is a multiple-value field, or a subdescriptor or superdescriptor derived from a multiple-value field. However, if the multiple value field is within a periodic group, the index is allowed but identifies the occurrence within the PE group and not within the multiple-value field.

D Daylight Savings Time Indicator
For fields with the time zone option TZ, a selection for daylight savings time can also be made using a D indicator element similar to that described in the section *Daylight Savings Indicator (D)*, in the format buffer description elsewhere in this guide.

In search buffers, the following rules apply:

- A daylight savings indicator cannot be specified without a corresponding date-time field. The date-time field must be defined with one of the following date-time edit masks: DATE-TIME, TIMESTAMP, or NATTIME).
- The daylight savings indicator must precede the corresponding date-time field. For example:

```
AAD,AA,14,U,E(DATETIME),GE.
```

Be sure to read *Daylight Savings Indicator Rules*, in the format buffer description elsewhere in this guide for more information about daylight savings indicator usage in format and search buffers.

S (Significance) and Null Indicators

For fields with the SQL null value compression option NC, a selection for "null" or "not null" can also be made using an "S" null indicator element similar to that described in the section *The SQL Significance Indicator and Field Series Notation*.

Note: The NC option cannot be applied to fields with the NU (null-value suppression), FI (fixed storage), MU (multiple-value), or PE (periodic group) options, or to group fields.

The SQL significance ("S") indicator must be added to the field name ("field-nameS") and the corresponding SQL null indicator must be specified in the value buffer.

The following hexadecimal null indicators are allowed as search argument values:

FFFF select null values

0000 select non-null values

Any other null indicator value causes an Adabas response code 52 (ADARSP052).

The null indicator (hexadecimal FFFF or 0000) has a standard length of two bytes and fixed-point format; this length and format cannot be overridden.

The "S" indicator can only be used with the equals (=) value-operator; using S with any other value operators causes an Adabas response code 61 (ADARSP061).

Examples:

The S significance operator is part of the search argument for the field AA.

```
AAS.
```

Select records with the FN field value of packed +1 and the AA field value of null (undefined):

Search Buffer	`FN  , 2 , P , D ,  AAS.`	Search argument
Value Buffer	`001F  FFFF`	Field value specification

Note: Insignificant null values are not stored in the index. This can cause a search-for-null operation to be quite costly for an application program's performance.

Select records with the FN field value of packed +1 and the AA field value of non-null:

Search Buffer	`FN  , 2 , P , D ,  AAS.`	Search argument
Value Buffer	`001F  0000`	Field value specification

length

The length of the field/descriptor value as provided in the value buffer. If the length is omitted, the value in the value buffer must comply with the standard length of the field/descriptor, as shown in ***Length and Data Format***.

format

The format of the field/descriptor value as provided in the value buffer. If the format is omitted, the value in the value buffer must comply with the standard format of the field/descriptor, as shown in ***Length and Data Format***.

value-operator

A value-operator indicates the logical operation to be performed between the preceding descriptor and its corresponding value in the value buffer.

The following operators may be specified:

Operator	Description
EQ (or) =	equals
GE	greater than or equal to
GT (or) >	greater than
LE	less than or equal to
LT (or) <	less than
NE	not equal to

If no value-operator is specified, an equals (EQ) operation is assumed.

Examples:

The following search buffer examples show the use of a value-operator:

Example	Description
AA.	AA equals the value specified in the value buffer (the default)
AA,LT.	AA is less than the value specified in the value buffer
AA,GE.	AA is greater than or equal to the value specified in the value buffer.

The following search buffer using the NE (not equal to) operator selects all records with the FN field not equal to "MIKE":

Search Buffer	`FN,4,A,NE.`	Search argument
Value Buffer	`MIKE`	Field value specification

Replacing the NE operator in this example with EQ (equal to) would select only records with FN field with values of "MIKE".

Connecting Search Expressions

A *connecting operator* may be used to connect search expressions. The permissible connecting operators are as follows:

Operator	Description
D	The results of two search expressions are to be combined using a logical AND operation. For example: `AA,D,AB.`

O	The results of two search expressions are to be combined using a logical OR operation. The OR operator may only be used to connect search expressions which use the same descriptor. Here is a valid and an invalid example:
	Valid:
	`AA,O,AA.`
	Invalid (two different descriptors are used):
	`AA,O,AB.`

R	Fields or command IDs that point to ISN lists derived from different descriptors are to be combined using a logical OR operation. For example: `AA,5,A,R,AB,LT.`

S	A FROM-TO range (inclusive) which involves two search expressions. The same descriptor must be used in both expressions. Here is a valid and an invalid example:
	Valid:
	`AA,S,AA.`
	Invalid (two different descriptors are used):
	`AA,S,AB. ↩`

N	Excludes a single value or a range of values from the immediately preceding FROM-TO range. This operator can only be specified in conjunction with the S operator, and must apply to the same

Operator	Description

	field specified in the FROM-TO range. Phonetic descriptors cannot be specified. Here are some valid and invalid examples:
	■ Valid:
	`AA,S,AA,N,AA.`
	Invalid (two different descriptors are used):
	`AA,S,AA,N,AB.`
	■ Valid:
	`AA,S,AA,N,AA,S,AA.`
	Invalid (two different descriptors are used):
	`AA,S,AA,N,AA,S,AB.`
	`AA,S,AA,N,AA,N,AB.`

Y	The results of any number of D, O, R, S, and N search operations can be combined using a logical AND operation. For example: `AA,D,AB,Y,AA,O,AA,Y,AA,S,AA,N,AA,S,AA.` The Y connecting operator functions like parentheses: only one level is allowed; that is, nested parentheses are not supported. All search expressions connected with the Y operator must apply to the same file.

If different operators are used within a single search buffer argument, the operators are processed in the following sequence:

1. Evaluate all S operations, as described in this documentation.
2. Evaluate all N operations, as described in this documentation.
3. Evaluate all O operations, as described in this documentation.
4. Evaluate all D operations, if needed.
5. Evaluate all R operations, if needed.
6. Evaluate all Y operations, if needed.

Example:

The following search buffer:

```
AA,S,AA,O,AA,D,AB,R,AC,D,AD.
```

is processed in this sequence:

```
( ( (AA,S,AA),O,AA),D,AB),R,(AC,D,AD)
```

The following search buffer:

```
AA,D,AB,Y,AA,O,AA,Y,AA,S,AA,N,AA,S,AA.
```

is processed in this sequence:

```
(AA,D,AB),Y,(AA,O,AA),Y,((AA,S,AA),N,(AA,S,AA))
```

Searching One File

The following syntax statement is relevant when searching fields in a single file:

```
search-expression [{,connecting-operator,search-expression}...] .
```

For the syntax of the `search-expression`, read ***Search Expression***, elsewhere in this section. For information about the `connecting-operator`, read ***Connecting Search Expressions***, elsewhere in this section.

Searching Multiple, Physically Coupled Files

The following syntax statement is relevant for multiple-file searches in which fields from two or more physically coupled files are to be used:

```
/file-x/search-expression[{,connecting-operator,search-expression}...]
{,D,/file-y search-expression/[{,connecting-operator,search-expression}...]}... .
```

where `file-x` and `file-y` are the file numbers of the physically coupled files. For information about search buffer syntax using physically coupled files, see ***Physically Coupled Files***, elsewhere in this section. For the syntax of the `search-expression`, read ***Search Expression***, elsewhere in this section. For information about the `connecting-operator`, read ***Connecting Search Expressions***, elsewhere in this section.

Searching One or More Files Using Soft Coupling

The following syntax statement is relevant for searching one or more files using soft coupling:

```
(m-file,m-field,s-file,s-field[{;m-file,m-field,s-file,s-field ↩
}...])/s-file-x/search-expression[{,connecting-operator,search-expression}... ]
[{,D,/s-file-y/search-expression[{,connecting-operator,search-expression}...]}...] .
```

where m-file and s-file are...., m-field `s-file-x` and `s-file-y` are the file numbers of the softly coupled files.

For information about search buffer syntax using soft coupling, see ***Soft Coupling***, elsewhere in this section. For the syntax of the `search-expression`, read ***Search Expression***, elsewhere in this section. For information about the `connecting-operator`, read ***Connecting Search Expressions***, elsewhere in this section.

Physically Coupled Files

The syntax of the search buffer for a multiple-file search in which fields from two or more physically coupled files are to be used is:

```
/file-x/search-expression[{,connecting-operator,search-expression}...]
{,D,/file-y search-expression/[{,connecting-operator,search-expression}...]}... .
```

The search criteria of the physically coupled files can be specified in any order. The ISN values actually returned are from the coupled file specified by the Adabas control block's file number field; this file is called the "primary" file.

The elements in this syntax are now described:

file-x **and** *file-y*
: The file numbers of the physically coupled files. All files specified must have been previously coupled using the COUPLE function of the ADAINV utility. A file number can appear only once for a given file. The file number must immediately precede its search criteria (consisting of one or more search expressions and appropriate connecting operators). A maximum of five (5) files may be specified in a single search buffer for physically coupled files.

D
: The only **connecting operator** allowed between the search criteria of the physically coupled files is the AND (D) symbol.

search-expression
: A search expression for the associated physically coupled file. For the syntax of the *search-expression*, read ***Search Expression***, elsewhere in this section.

connecting-operator
: A connecting operator to connect the search expressions of the search criteria for an individual physically coupled file. While the connecting operator between search criteria for the physically coupled files must be "D" (AND), the connecting operators between the search expressions that comprise the search criteria for an individual file can be any of the operators described in ***Connecting Search Expressions***, elsewhere in this section.

Example:

Find the ISNs of all the records in file 1 that contain the three-byte (length override) unpacked decimal (format override) value "+20" in their AB fields, and that are coupled to records in file 2 containing the value "ABCDE" for field RB, which has a standard length of ten bytes and an alpha-numeric format.

Search Buffer	`/1/AB,3,U,D,/2/RB.`	Search argument

Value Buffer	character-notation
	`020ABCDEbbbbb`
	hex-notation
	`F0F2F0C1C2C3C4C54040404040`
		Field value specification

Soft Coupling

The syntax of search buffer for a search in which soft coupling is to be used is:

```
(m-file,m-field,s-file,s-field[{;m-file,m-field,s-file,s-field ↩
}...])/s-file-x/search-expression[{,connecting-operator,search-expression}... ]
[{,D,/s-file-y/search-expression[{,connecting-operator,search-expression}...]}...] .
```

The elements in this syntax are now described:

m-file, *m-field*

For *m-file*, specify the number of the main file. This file must also be specified in the file number field of the Adabas control block. The final resulting ISN list will include ISNs contained in the main file only.

For *m-field*, specify the field in the main file that is to be used as the soft-coupling link field. This field must be a descriptor, subdescriptor, superdescriptor, or hyperdescriptor. It may not be a long alphanumeric field or be contained within a periodic group.

The combination of *m-file*, *m-field*, *s-file*, and *s-field* specifications comprise a single soft coupling. A maximum of 42 soft-coupling criteria may be specified. All of the soft coupling must be specified in one set of parentheses.

s-file, *s-field*

For *s-file*, specify the file number of the search file; for *s-field*, specify a field within the search file. For each ISN selected from this search file (according to the search criteria), the field specified as *s-field* will be read. The value of the field will then be used to determine which ISNs in the main file have a matching value.

The field may be a descriptor or nondescriptor; it can be a subdescriptor, superdescriptor, hyperdescriptor, or a long alphanumeric field. It must have the same format as the corresponding *m-field*. The standard length may be different. The field may not be contained within a periodic group.

The combination of *m-file*, *m-field*, *s-file*, and *s-field* specifications comprise a single soft coupling. A maximum of 42 soft-coupling criteria may be specified. All of the soft coupling must be specified in one set of parentheses.

`s-file-x` **and** `s-file-y`
: The file number of the coupled files for which you want to specify search criteria. A file number can appear only once for a given file. The file number must immediately precede its search criteria (consisting of one or more search expressions and appropriate connecting operators). A maximum of five (5) files may be specified in a single search buffer.

D
: The only **connecting operator** allowed between the search criteria of the coupled files is the AND (D) symbol.

`search-expression`
: A search expression for the associated coupled file. For the syntax of the `search-expression`, read ***Search Expression***, elsewhere in this section.

`connecting-operator`
: A connecting operator to connect the search expressions of the search criteria for an individual coupled file. While the connecting operator between search criteria for the physically coupled files must be "D" (AND), the connecting operators between the search expressions that comprise the search criteria for an individual file can be any of the operators described in ***Connecting Search Expressions***, elsewhere in this section.

Value Buffers

The search and value buffers are used together to define:

- the search criteria to select a set of records using a FIND command (S1, S2, S4); and
- the range of values to be traversed by logical sequential read commands (L3/6, L9).

If a value buffer is provided, a search buffer is also expected. If it is not provided, Adabas will create a dummy search buffer to pair with the value buffer.

Only one search and value buffer pair should be specified in a single Adabas direct call.

The user provides the search expressions in the search buffer and the values which correspond to the search expressions in the value buffer.

In the value buffer, the user specifies the values for each descriptor specified in the search buffer.

If the search expression is a command ID, no corresponding entry is made in the value buffer.

The values provided must be in the same sequence as the corresponding search expressions specified in the search buffer. All values provided must correspond to the standard length and format of the corresponding descriptor unless the user has explicitly overridden the standard length or format in the search buffer.

No intervening blanks or other characters such as a comma can be inserted between values in the value buffer. A period is not required to end the value buffer entry.

- SQL Null Values and Indicators
- Sign Handling

SQL Null Values and Indicators

When searching for fields defined with the NC (SQL null not counted) option, the search buffer field definition must contain a null significance (S) indicator and the corresponding value buffer argument value must display a two-byte binary null value indicator. See the section ***S (Significance) and Null Indicators*** for more information and examples of the null value indicator in the value buffer.

Sign Handling

Binary values are treated as unsigned numbers. Fixed-point, unpacked, and packed values are treated as signed numbers. Valid signs which may be provided are described in thissection:

Fixed Value Signs

For fixed values, the sign is contained in bit 0 (high-order bit):

- 0 = positive
- 1 = negative (two's complement)

Here are two fixed value sign examples showing the hexadecimal notation and the decimal equivalent:

```
00000005 = +5
FFFFFFFB = -5
```

Unpacked Value Signs

For unpacked values, the sign is contained in the four high-order bits of the low-order byte:

- C or A or F or E = positive (CAFE)
- B or D = negative (BD)

Here are two unpacked value sign examples showing the hexadecimal notation and the decimal equivalent:

```
F1F2F3  = +123
F1F2D3  = -123
```

Packed Value Signs

For packed values, the sign is contained in the four low-order bits of the low-order byte:

- A or C or E or F = positive
- B or D = negative

If a search value is being provided for a superdescriptor which is derived from a packed field, an F positive sign or a D negative sign must be provided.

Here are two packed value sign examples showing the hexadecimal notation and the decimal equivalent:

```
X'123F' = +123
X'123C' = +123
X'123D' = -123
```

ISN Buffers

The ISN buffer defines an area in storage to which Adabas can return the ISNs of the records that satisfy the specified search criteria for a command. In addition, if `Command Option 1` for a command is set to "M" (or "P"), assuming these are valid settings for that command, *and* if the command is issued using the ACB direct call interface, the ISN buffer holds the record descriptor elements (RDEs) of multifetched or prefetched records awaiting processing. If, instead, the command is issued using the ACBX direct call interface, the ISN buffer is not used for this purpose; the multifetch buffer is used instead.

When needed, only one ISN buffer should be specified in a direct call.

The four-byte binary ISNs are usually provided in the ISN buffer in ascending sequence. For the S2 or S9 command, the ISNs are provided according to the user-specified sort sequence. If the ISN buffer is not large enough to contain the entire resulting ISN list, Adabas stores the overflow ISNs on the Adabas work data set, if requested. These overflow ISNs can then be retrieved at a later time.

If the resulting ISNs are to be read using the GET NEXT option of the L1 or L4 commands, the ISN buffer is not needed.

The ISN buffer also supplies an ISN list to be used as input when the ET or BT command specifies a `Command Option 1` of "M" (or "P").

7 Adabas Commands

- Adabas Command ACBs 50
- ACB Command Buffers 52
- Adabas Command ACBXs 52
- ACBX Command ABDs and Buffers 52
- A1 Command -- Update Record 52
- BT Command -- Back Out Transaction 55
- C1 Command -- Record Checkpoint 57
- C3 Command -- Write Checkpoint 59
- C5 Command -- Write User Data to PLOG 61
- CL Command -- Close Session 64
- E1 Command -- Delete Record 66
- ET Command -- End Transaction 68
- HI Command -- Hold Record 71
- L1/L4 Commands -- Read Record 73
- L2/L5 Commands -- Read Physical Sequential Record 75
- L3/L6 Commands -- Read Logical Sequential Record 78
- L9 Command -- Read Descriptor Values 81
- LF Command -- Read Field Definitions 84
- N1/N2 Commands -- Add New Record 86
- OP Command -- Open User Session 89
- RC Command -- Release Command ID or Global Format ID 92
- RE Command -- Read ET User Data 94
- RI Command -- Release Held Record 96
- S1/S2/S4 Commands -- Find Records 98
- S5 Command -- Find Coupled ISNs 101
- S8 Command -- Process ISN Lists 103
- S9 Command -- Sort ISN List 105

Each Adabas command is described in the table below; select a command to see detailed control block and buffer information for it. In addition, the following summary tables are provided showing the use of the ACB, ACBX, and buffers by all Adabas commands:

- *Adabas Command ACBs*
- *ACB Command Buffers*
- *Adabas Command ACBXs*
- *ACBX Command ABDs and Buffers*

Note: We recommend that you set unused ACB and ACBX fields to binary zeros before a direct call is initiated.

Command Code	Summary	Description
A1	Update record	Update record(s) (hold option)
BT	Backout transaction	Remove database updates for ET logic users
C1	Write checkpoint	Write command ID, PLOG, RABN RABN checkpoint, buffer flush option
C3	Write SYNX-03 checkpoint	Write SYNX-03 checkpoint for exclusive control update users; option to store user data
C5	Write user data to protection log	Write user data on SIBA/PLOG
CL	Close user session	End ET session and update database
E1	Delete record / refresh file	Delete record (hold option) or refresh file
ET	End transaction	End and save current transaction
HI	Hold record	Prevent record update by other users
L1	Read record	Read record of specified ISN
L2	Read physical sequential record	Read records in physical order
L3	Read logical sequential record	Read records in descriptor value order
L4	Read and hold record	Read record and hold, "wait for held record/issue return code" option
L5	Read physical sequential record and hold	Read records in physical order and hold, "wait/issue return code" option
L6	Read logical sequential record and hold	Read records in descriptor value order with "wait/issue return code" option
L9	Read descriptor values	Read the values of a specified descriptor
LF	Read field definition	Read the characteristics of all fields in a file
N1	Add record with Adabas-assigned ISN	Add new database record with ISN assigned by Adabas
N2	Add record with user-assigned ISN	Add new database record with user-assigned ISN
OP	Open user session	Open user session

Command Code	Summary	Description
RC	Release command ID or global format ID	Release one or more command IDs or a global format ID for the issuing user
RE	Read ET user data	Read ET data for this, another, or all users
RI	Release held record and ISN	Release held record and ISN
S1	Find records	Return count and ISNs of records satisfying the search criterion
S2	Find records in user-specified order	Return count of records and ISNs in user-specified order
S4	Find records and hold	Return count and ISNs of records satisfying the search criterion and put first ISN in list on hold
S5	Find coupled ISNs	Return or save a list of coupled ISNs for the specified file
S8	Process ISN lists	Combine two ISN lists from the same file with an AND, OR, or NOT operation
S9	Sort ISN lists	Sort ISN list in ascending ISN or descriptor-specified sequence

Adabas includes some V* and Y* commands, which you may see mentioned in Adabas shutdown statistics or in Adabas Online System (AOS) screens. These commands are used internally by Adabas and Adabas add-on products and should not be used in direct calls in your applications. Should you use them, errors will result.

Adabas Command ACBs

ACB Command Buffers

Adabas Command ACBXs

ACBX Command ABDs and Buffers

A1 Command -- Update Record

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric / binary	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	U
ISN Lower Limit	17-20	binary	--	A[1]
ISN Quantity	21-24	binary	--	A[1]
Format Buffer Length	25-26	binary	F	U
Record Buffer Length	27-28	binary	F	U
	29-34	--	--	--
Command Option 1 / 2	35-36	alphanumeric	F	U
	37-44	--	--	--
Additions 2	45-48	alphanumeric / binary		A
Additions 3	49-56	alphanumeric	F	A
Additions 4	57-64	alphanumeric	F	A

Field	Position	Format	Before Adabas Call	After Adabas Call
Additions 5	65-72	alphanumeric	F	U
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Notes

These fields are used and not reset by Adabas if coupled files are used.

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	F	U

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	binary	---	---
Version Indicator	3-4	binary	F	U
	5-6	binary	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	binary	---	---
Response Code	11-12	binary	---	U
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	U
	33-36	---	---	---
ISN Lower Limit	37-40	binary	---	A[1]
	41-44	---	---	---

Field	Position	Format	Before Adabas Call	After Adabas Call
ISN Quantity	45-48	binary	---	A[1]
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
	51-68	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
Additions 4	77-84	alphanumeric	F	A
Additions 5	85-92	alphanumeric/ binary	F	U
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-128	---	---	---
Compressed Record Length	129-136	binary	---	A
Decompressed Record Length	137-144	binary	---	A
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
	169-193	---	---	---

Notes

These fields are used and not reset by Adabas if coupled files are used.

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

--- Not used

BT Command -- Back Out Transaction

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	binary	--	A
File Number *	9-10	binary	F *	U
Response Code	11-12	binary	--	A
	13-16	--	--	--
ISN Lower Limit	17-20	binary	F	U
	21-32	--	--	--
ISN Buffer Length **	33-34	binary	F **	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
	37-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
ISN **	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Required only if Command Option 2 is specified

** Required only if Command Option 1 is specified

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	binary	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	---	A
Database ID*	17-20	numeric	F*	U
File Number*	21-24	numeric	F*	U
	25-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	45-48	---	---	---
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
Command Option 3	51	alphanumeric	F	U
	52-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
	169-193	---	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
ISN **	F	U

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
* Required only if Command Option 2 is specified
** Required only if Command Option 1 is specified
--- Not used

C1 Command -- Record Checkpoint

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	A
File Number *	9-10	binary	F	U
Response Code	11-12	binary	--	A
	13-34	--	--	--
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
	37-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

None used.

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	A
Database ID*	17-20	numeric	F	U
	21-48	---	---	---
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
	51-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
	169-193	---	---	---

ABDs and Buffers

None used.

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

--- Not used

C3 Command -- Write Checkpoint

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
	5-8	--	--	--
File Number **	9-10	binary	F	U
Response Code	11-12	binary	--	A
	13-26	--	--	--
Record Buffer Length	27-28	binary	F	U
	29-35	--	--	--
Command Option 2	36	alphanumeric	F	U
	37-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	*	
Record	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but must be included in parameter list of call statement

** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
---	13-16	---	---	---
Database ID**	17-20	numeric	F	U
	21-49	---	---	---
Command Option 2	50	alphanumeric	F	U
	51-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
	169-193	---	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	*	
Record	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but should be included in Adabas call or one will be automatically generated.

** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

C5 Command -- Write User Data to PLOG

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
	5-8	--	--	--
File Number **	9-10	binary	F	U
Response Code	11-12	binary	--	A
	13-26	--	--	--
Record Buffer Length	27-28	binary	F	U
	29-34	--	--	--
Command Option 1***	35	alphanumeric	F	U
	36	--	--	--
Additions 1***	37-44	alphanumeric	F	U

Field	Position	Format	Before Adabas Call	After Adabas Call
	45-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	*	
Record	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but must be included in parameter list of call statement

** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

However, if you are using Event Replicator for Adabas and the Command Option 1 field is set to "R", a file number must be specified to identify the file to which the C5 command applies. For more information, read your Event Replicator for Adabas documentation.

*** Only used if you are using Event Replicator for Adabas. For more information, read your Event Replicator for Adabas documentation.

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
	13-16	---	---	---
Database ID**	17-20	numeric	F	U
File Number***	21-24	numeric	F	U

Field	Position	Format	Before Adabas Call	After Adabas Call
	25-48	---	---	---
Command Option 1***	49	alphanumeric	F	U
	50-56	---	---	---
Additions 1***	57-64	alphanumeric/ binary	F	U
	65-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	*	
Record	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but should be included in Adabas call or one will be automatically generated.

** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

However, if you are using Event Replicator for Adabas and the Command Option 1 field is set to "R", a file number must be specified to identify the file to which the C5 command applies. For more information, read your Event Replicator for Adabas documentation.

*** Only used if you are using Event Replicator for Adabas. For more information, read your Event Replicator for Adabas documentation.

--- Not used

CL Command -- Close Session

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	binary	--	A
File Number ***	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	--	A
ISN Lower Limit	17-20	binary	--	A
ISN Quantity	21-24	binary	--	A
	25-26	--	--	--
Record Buffer Length	27-28	binary	F *	U
	29-35	--	--	--
Command Option 2	36	alphanumeric	F	U
	37-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format *	**	--
Record *	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Required only if user data is to be stored

** Not used but must be included in parameter list of call statement if user data to be stored

*** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	---	A
Database ID***	17-20	numeric	F	U
	21-28	---	---	---
Number of I/Os	29-32	binary	---	A
	33-36	---	---	---
Number of Commands	37-40	binary	---	A
CPU Time	41-48	binary	---	A
	49	---	---	---
Command Option 2	50	alphanumeric	F	U
	51-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format *	**	--
Record *	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Required only if user data to be stored

** Not used but should be included in Adabas call or one will be automatically generated.

*** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

E1 Command -- Delete Record

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric / binary	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	U
ISN Lower Limit	17-20	binary	--	A[1]
ISN Quantity	21-24	binary	--	A[1]
	25-34	--	--	--
Command Option 1	35	alphanumeric	F	U

Field	Position	Format	Before Adabas Call	After Adabas Call
	36-48	--	--	--
Additions 3	49-56	alphanumeric	F	A
Additions 4	57-64	alphanumeric	F	A
	65-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Notes

These fields are used and not reset by Adabas if coupled files are used.

Buffer Areas

None used.

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28		---	---
ISN	29-32	binary	F	U
	33-36		---	---
ISN Lower Limit	37-40	binary	---	A[1]
	41-44		---	---

Field	Position	Format	Before Adabas Call	After Adabas Call
ISN Quantity	45-48	binary	---	A[1]
Command Option 1	49	alphanumeric	F	U
	50-68	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
Additions 4	77-84	alphanumeric	F	A
	85-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

Notes

These fields are used and not reset by Adabas if coupled files are used.

ABDs and Buffers

None used.

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
--- Not used

ET Command -- End Transaction

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure

- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	binary	--	A
File Number****	9-10	binary	F	U
Response Code	11-12	binary	--	A
	13-16	--	--	--
ISN Lower Limit	17-20	binary	F	U
ISN Quantity	21-24	binary	--	A
	25-26	--	--	
Record Buffer Length	27-28	binary	F *	U
	29-32	--	--	--
ISN Buffer Length **	33-34	binary	F **	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
	37-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format *	***	--
Record *	F	U
ISN **	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Required only if ET data is to be stored

** Required for hold ISN option; optional for multifetch option

*** Not used but must be included in parameter list of call statement if ET data is to be stored

**** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	---	A
Database ID****	17-20	numeric	F	U
	21-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	41-44	---	---	---
ISN Quantity	45-48	binary	---	A
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
Command Option 3	51	alphanumeric	F	U
	52-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format *	***	--
Record *	F	U
ISN **	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Required only of ET data to be stored

** Required for hold ISN option; optional for multifetch option

*** Not used but should be included in Adabas call if ET data is to be stored. If not specified, one will be automatically generated.

**** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

--- Not used

HI Command -- Hold Record

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
	5-8	--	--	--
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	U
	17-34	--	--	--
Command Option 1	35	alphanumeric	F	U
	36-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

None used.

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
---	13-16	---	---	---
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	U
	33-48	---	---	---
Command Option 1	49	alphanumeric	F	U
	50	---	---	---
Command Option 3	51	alphanumeric	F	U
	52-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

Buffer Areas

None used.

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

-- Not used

L1/L4 Commands -- Read Record

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	U *
ISN Lower Limit	17-20	binary	F	U
ISN Quantity	21-24	binary	F*	U
Format Buffer Length	25-26	binary	F	U
Record Buffer Length	27-28	binary	F	U
	29-32	--	--	--
ISN Buffer Length **	33-34	binary	F	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
	37-44	--	--	--
Additions 2	45-48	binary / binary	--	A
Additions 3	49-56	alphanumeric	F	A
Additions 4	57-64	alphanumeric	F	A
Additions 5	65-72	alphanumeric	F	U
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	--	A
ISN **	--	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Except for special options

** The ISN buffer and length are required only if the multifetch or prefetch option is specified

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	U*
	33-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	41-44	---	---	---
ISN Quantity	45-48	binary	F*	U
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
Command Option 3 (L4 only)	51	alphanumeric	F	U
	52-64	---	---	---

Field	Position	Format	Before Adabas Call	After Adabas Call
Additions 3	69-76	alphanumeric/ binary	F	A
Additions 4	77-84	alphanumeric	F	A
Additions 5	85-92	alphanumeric/ binary	F	U
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-128	---	---	---
Compressed Record Length	129-136	binary	---	A
Decompressed Record Length	137-144	binary	---	A
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	--	A
Multifetch **	--	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Except for special options

** The multifetch buffer is required only if the multifetch option is specified.

--- Not used

L2/L5 Commands -- Read Physical Sequential Record

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure

- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	A
ISN Lower Limit	17-20	binary	F	U
	21-24	--	--	--
Format Buffer Length	25-26	binary	F	U
Record Buffer Length	27-28	binary	F	U
	29-32	--	--	--
ISN Buffer Length *	33-34	binary	F	U
Command Option 1	35	alphanumeric	F	U
	36-44	--	--	--
Additions 2	45-48	binary / binary	--	A
Additions 3	49-56	alphanumeric	F	A
Additions 4	57-64	alphanumeric	F	A
Additions 5	65-72	alphanumeric	F	U
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	F**	U
Record	--	A
ISN *	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* The ISN buffer and length are required only of the multifetch or prefetch option is specified

** May contain compress option control characters "C."

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	A
	33-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	41-48	---	---	---
Command Option 1	49	alphanumeric	F	U
	50	---	---	---
Command Option 3 (L5 only)	51	alphanumeric	F	U
	52-64	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
Additions 4	77-84	alphanumeric	F	A
Additions 5	85-92	alphanumeric/ binary	F	U
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-128	---	---	---
Compressed Record Length	129-136	binary	---	A
Decompressed Record Length	137-144	binary	---	A
Command Time	145-152	binary	---	A

Field	Position	Format	Before Adabas Call	After Adabas Call
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	F**	U
Record	--	A
Multifetch *	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* A multifetch buffer is required only if the multifetch option is specified.

** May contain compress option control characters "C."

--- Not used

L3/L6 Commands -- Read Logical Sequential Record

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	binary	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	A
ISN Lower Limit	17-20	binary	F	U

Field	Position	Format	Before Adabas Call	After Adabas Call
	21-24	--	--	--
Format Buffer Length	25-26	binary	F	U
Record Buffer Length	27-28	binary	F	U
Search Buffer Length	29-30	binary	F *	U
Value Buffer Length	31-32	binary	F *	U
ISN Buffer Length **	33-34	binary	F	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
Additions 1	37-44	alphanumeric	F	A
Additions 2	45-48	binary / binary	--	A
Additions 3	49-56	alphanumeric	F	A
Additions 4	57-64	alphanumeric	F	A
Additions 5	65-72	alphanumeric	F	U
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	F ***	U
Record	--	A
Search *	F	U
Value *	F	U
ISN **	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Required only if value start option is being used

** The ISN buffer and length is required only if the multifetch or prefetch option is specified.

*** May contain compress option control characters "C."

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	A
	33-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	41-48	---	---	---
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
Command Option 3 (L6 only)	51	alphanumeric	F	U
	52-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F	A
Additions 3	69-76	alphanumeric/ binary	F	A
Additions 4	77-84	alphanumeric	F	A
Additions 5	85-92	alphanumeric/ binary	F	U
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-128	---	---	---
Compressed Record Length	129-136	binary	---	A
Decompressed Record Length	137-144	binary	---	A
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	F ***	U
Record	--	A
Search *	F	U
Value *	F	U
Multifetch **	F	U

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
* Required only if value start option is being used
** The multifetch buffer is required only if the multifetch option is specified.
*** May contain compress option control characters "C."
--- Not used

L9 Command -- Read Descriptor Values

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	--	A
ISN Lower Limit	17-20	binary	F	A
ISN Quantity	21-24	binary	--	A

Field	Position	Format	Before Adabas Call	After Adabas Call
Format Buffer Length	25-26	binary	F	U
Record Buffer Length	27-28	binary	F	U
Search Buffer Length	29-30	binary	F	U
Value Buffer Length	31-32	binary	F	U
ISN Buffer Length *	33-34	binary	F	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
Additions 1	37-44	alphanumeric	F	U
Additions 2	45-48	binary	--	A
Additions 3	49-56	alphanumeric	F	A
	57-64	--	--	--
Additions 5	65-72	alphanumeric	F	U
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	--	A
Search	F	U
Value	F	U
ISN *	--	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* The ISN buffer and length required only if the multifetch or prefetch option is specified

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	---	A
	33-36	---	---	---
ISN Lower Limit	37-40	binary	F	A
	41-44	---	---	---
ISN Quantity	45-48	binary	---	A
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
	51-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F	U
---	65-68	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
	77-84	---	---	---
Additions 5	85-92	alphanumeric/ binary	F	U
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-128	---	---	---
Compressed Record Length	129-136	binary	---	A
Decompressed Record Length	137-144	binary	---	A
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	---	A
Search	F	U
Value	F	U
Multifetch *	---	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* The multifetch buffer is required only if the multifetch option is specified

--- Not used

LF Command -- Read Field Definitions

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
	5-8	--	--	--
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
	13-26	--	--	--
Record Buffer Length	27-28	binary	F	U
	29-35	--	--	--
Command Option 2	36	alphanumeric	F	U
	37-48	--	--	--

Field	Position	Format	Before Adabas Call	After Adabas Call
Additions 3	49-56	alphanumeric	F	A
	57-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	*	--
Record	--	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but must be included in parameter list of call statement

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
	13-16	---	---	---
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-49	---	---	---
Command Option 2	50	alphanumeric	F	U
	51-68	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
	77-114	---	---	---
Error Subcode	115-116	binary	---	A

Field	Position	Format	Before Adabas Call	After Adabas Call
	117-128	---	---	---
Compressed Record Length	129-136	binary	---	A
Decompressed Record Length	137-144	binary	---	A
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	*	--
Record	--	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but should be included in Adabas call or one will be automatically generated.

-- Not used

N1/N2 Commands -- Add New Record

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A

Field	Position	Format	Before Adabas Call	After Adabas Call
ISN	13-16	binary	F	A/U[1]
ISN Lower Limit	17-20	binary	--	A[2]
ISN Quantity	21-24	binary	--	A[2]
Format Buffer Length	25-26	binary	F	U
Record Buffer Length	27-28	binary	F	U
	29-44	--	--	--
Additions 2	45-48	alphanumeric	--	A
Additions 3	49-56	alphanumeric	F	A
Additions 4	57-64	alphanumeric	F	A
Additions 5	65-72	alphanumeric	F	U
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Notes

1. Supplied by Adabas for N1; unchanged for N2.
2. These fields are used and not reset by Adabas if coupled files are used.

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	A/U[1]
	33-36	---	---	---
ISN Lower Limit	37-40	binary	---	A[2]
	41-44	---	---	---
ISN Quantity	45-48	binary	---	A[2]
	49-64	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
Additions 4	77-84	alphanumeric	F	A
Additions 5	85-92	alphanumeric/ binary	F	U
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-128	---	---	---
Compressed Record Length	129-136	binary	---	A
Decompressed Record Length	137-144	binary	---	A
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

Notes

1. Supplied by Adabas for N1; unchanged for N2.
2. These fields are used and not reset by Adabas if coupled files are used.

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	F	U

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

--- Not used

OP Command -- Open User Session

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	--	A
File Number**	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	--	A
ISN Lower Limit	17-20	binary	F	A
ISN Quantity	21-24	binary	F	A
	25-26	--	--	--
Record Buffer Length	27-28	binary	F	U
	29-34	--	--	--
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
Additions 1	37-44	alphanumeric	F	U

Field	Position	Format	Before Adabas Call	After Adabas Call
Additions 2	45-48	alphanumeric	--	A
	49-56	--	--	--
Additions 4	57-64	binary	F	A
Additions 5	65-72	binary	--	A
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	*	
Record	F	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but must be included in parameter list of the call statement

** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	---	A
Database ID**	17-20	numeric	F	U
	21-28	---	---	---
ISN	29-32	binary	---	A
	33-36	---	---	---

Field	Position	Format	Before Adabas Call	After Adabas Call
ISN Lower Limit	37-40	binary	F	A
	41-44	---	---	---
ISN Quantity	45-48	binary	F	A
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
	51-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F	U
Additions 2	65-68	binary	---	A
	69-76	---	---	---
Additions 4	77-84	alphanumeric	F	A
Additions 5	85-92	alphanumeric/ binary	---	A
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	*	
Record	F	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but should be included in Adabas call or one will be automatically generated.

** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

RC Command -- Release Command ID or Global Format ID

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
	13-34	--	--	--
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
Additions 1	37-44	alphanumeric	F	U
	45-64	--	--	--
Additions 5	65-72	alphanumeric	F	U
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

None used.

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-48	---	---	---
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
	51-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F	U
	65-84		---	---
Additions 5	85-92	alphanumeric/ binary	F	U
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

None used.

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

-- Not used

RE Command -- Read ET User Data

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	binary	--	A
File Number ****	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	A
	17-26	--	--	--
Record Buffer Length	27-28	binary	F	U
	29-34	--	--	--
Command Option 1	35	alphanumeric	F	U
	36	--	--	--
Additions 1	37-44	alphanumeric	F *	U/A **
Additions 2	45-48	alphanumeric	--	A
	49-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	***	--
Record	--	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Supplied ET data user ID when Command Option 1 equals "I"

** User ID for ET data in record buffer if Command Option 1 equals "A"

*** Not used but must be included in parameter list of call statement

**** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	---	A
Database ID****	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	A
	33-48	---	---	---
Command Option 1	49	alphanumeric	F	U
	50-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F*	U/A**
Additions 2	65-68	binary	---	A
	69-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	***	--
Record	--	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Supplied ET data user ID when Command Option 1 equals "I"

** User ID for ET data in record buffer if Command Option 1 equals "A"

*** Not used but should be included in Adabas call or one will be automatically generated.

**** A database ID is only necessary if you are accessing a database other than the application's default database (read in by ADARUN DBID parameter, provided in the loaded link globals table, or linked with the link routine).

--- Not used

RI Command -- Release Held Record

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
	5-8	--	--	--
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	U
	17-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

None used.

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
	13-16	---	---	---
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	U
	33-50	---	---	---
Command Option 3	51	alphanumeric	F	U
	52-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

None used.

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

--- Not used

S1/S2/S4 Commands -- Find Records

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	--	A
ISN Lower Limit	17-20	binary	F	U
ISN Quantity	21-24	binary	F *	A
Format Buffer Length	25-26	binary	F	U
Record Buffer Length	27-28	binary	F	U
Search Buffer Length	29-30	binary	F	U
Value Buffer Length	31-32	binary	F	U
ISN Buffer Length	33-34	binary	F	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
Additions 1	37-44	alphanumeric	F	U
Additions 2	45-48	binary / binary	--	A
Additions 3	49-56	alphanumeric	F	A
Additions 4	57-64	alphanumeric	F	A
Additions 5	65-72	alphanumeric	F	U
Command Time	73-76	binary	--	A

Field	Position	Format	Before Adabas Call	After Adabas Call
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	--	A
Search	F	U
Value	F	U
ISN	--	A

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
* Optional timeout value, in seconds
-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	---	A
	33-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	41-44	---	---	---
ISN Quantity	45-48	binary	F*	A

Field	Position	Format	Before Adabas Call	After Adabas Call
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
Command Option 3 (S4 only)	51	alphanumeric	F	U
	52-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F	U
Additions 3	69-76	alphanumeric/ binary	F	A
Additions 4	77-84	alphanumeric	F	A
Additions 5	85-92	alphanumeric/ binary	F	U
	93-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-128	---	---	---
Compressed Record Length	129-136	binary	---	A
Decompressed Record Length	137-144	binary	---	A
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
Format	F	U
Record	---	A
Search	F	U
Value	F	U
ISN	---	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Optional timeout value, in seconds

--- Not used

S5 Command -- Find Coupled ISNs

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	F	A
ISN Lower Limit	17-20	binary	F	U
ISN Quantity	21-24	binary	--	A
	25-32	--	--	--
ISN Buffer Length	33-34	binary	F	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
Additions 1	37-44	alphanumeric	F	U
	45-48	--	--	--
Additions 3	49-56	alphanumeric	F	A
	57-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	*	--
Record	*	--
Search	*	--
Value	*	--
ISN	--	A

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
* Not used but must be included in parameter list of call statement
-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	F	A
	33-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	41-44	---	---	---
ISN Quantity	45-48	binary	---	A
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
	51-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F	U
	65-68	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
	77-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
ISN	---	A

where:

F Supplied by user before Adabas call
A Supplied by Adabas
U Unchanged after Adabas call
--- Not used

S8 Command -- Process ISN Lists

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure
- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	--	A
ISN Lower Limit	17-20	binary	F	U
ISN Quantity	21-24	binary	--	A
	25-32	--	--	--
ISN Buffer Length	33-34	binary	F	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
Additions 1	37-44	alphanumeric	F	U
	45-48	--	--	--
Additions 3	49-56	alphanumeric	F	A

Field	Position	Format	Before Adabas Call	After Adabas Call
	57-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	*	--
Record	*	--
Search	*	--
Value	*	--
ISN	--	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but must be included in parameter list of call statement

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	---	A
	33-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	41-44	---	---	---

Field	Position	Format	Before Adabas Call	After Adabas Call
ISN Quantity	45-48	binary	---	A
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
	51-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F	U
	65-68	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
	77-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
ISN	---	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

--- Not used

S9 Command -- Sort ISN List

We recommend that you set unused ACB and ACBX fields to binary zeros before the direct call is initiated.

- ACB Control Block Structure

- ACBX Control Block Structure

ACB Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	--	--	--
Command Code	3-4	alphanumeric	F	U
Command ID	5-8	alphanumeric	F	U
File Number	9-10	binary	F	U
Response Code	11-12	binary	--	A
ISN	13-16	binary	--	A
ISN Lower Limit	17-20	binary	F	U
ISN Quantity	21-24	binary	F	A
	25-32	--	--	--
ISN Buffer Length	33-34	binary	F	U
Command Option 1	35	alphanumeric	F	U
Command Option 2	36	alphanumeric	F	U
Additions 1	37-44	alphanumeric	F	U
	45-48	--	--	--
Additions 3	49-56	alphanumeric	F	A
Additions 4	57-64	alphanumeric	F	A
	65-72	--	--	--
Command Time	73-76	binary	--	A
User Area	77-80	--	--	U

Buffer Areas

Buffer	Before Adabas Call	After Adabas Call
Format	*	--
Record	*	--
Search	*	--
Value	*	--
ISN	F	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

* Not used but must be included in parameter list of call statement

-- Not used

ACBX Control Block Structure

Field	Position	Format	Before Adabas Call	After Adabas Call
	1-2	---	---	---
Version Indicator	3-4	binary	F	U
	5-6	---	---	---
Command Code	7-8	alphanumeric	F	U
	9-10	---	---	---
Response Code	11-12	binary	---	A
Command ID	13-16	alphanumeric/ binary	F	U
Database ID	17-20	numeric	F	U
File Number	21-24	numeric	F	U
	25-28	---	---	---
ISN	29-32	binary	---	A
	33-36	---	---	---
ISN Lower Limit	37-40	binary	F	U
	41-44	---	---	---
ISN Quantity	45-48	binary	F	A
Command Option 1	49	alphanumeric	F	U
Command Option 2	50	alphanumeric	F	U
	51-56	---	---	---
Additions 1	57-64	alphanumeric/ binary	F	U
	65-68	---	---	---
Additions 3	69-76	alphanumeric/ binary	F	A
Additions 4	77-84	alphanumeric	F	A
	85-114	---	---	---
Error Subcode	115-116	binary	---	A
	117-144	---	---	---
Command Time	145-152	binary	---	A
User Area	153-168	not applicable	---	U
---	169-193	do not touch	---	---

ABDs and Buffers

ABD and Buffer	Before Adabas Call	After Adabas Call
ISN	F	A

where:

F Supplied by user before Adabas call

A Supplied by Adabas

U Unchanged after Adabas call

--- Not used

Index

B

bold, 1
braces ({}), 1
brackets ([]), 1

C

C indicator, 24
choices in syntax, 1
connecting operators, 39

D

D indicator, 24

E

ellipsis (...), 2

F

format buffer
- asterisk length notation, 27
- count indicator (C), 24
- daylight savings indicator (D), 24
- edit mask notation, 28
- field length and data format, 25
- field name specifications, 23
- field series notation, 29
- field syntax, 23
- highest occurrence / value indicator (N), 25
- index or range notation, 23
- length indicator (C), 25
- space notation (nX), 30
- SQL significance indicator (S), 25
- text insertion notation, 30

I

indentation, 2
italic, 1

L

L indicator, 25
lowercase, 1

N

N indicator, 25
normal font, 1

O

optional syntax elements, 1

P

punctuation and symbols in syntax, 2

R

record buffer
- alpha field with long alpha (LA) option, 34
- SQL null indicator, 32

repeated syntax elements, 2
required syntax elements, 1

S

S indicator, 25
search buffer
- daylight savings time indicator, 36
- search expression, 35
- search expression and occurrence index, 36
- search expression and significance and null indicators, 37
- search expression connecting operators, 39
- search expression field formats, 38
- search expression field lengths, 38
- search expression value operators, 38
- search expressions and field types, 35
- syntax, 35
- using physically coupled files, 42
- using soft coupling, 43

subparameter syntax, 2
superdescriptor
- as a single-file search argument, 36

Syntax conventions
- indentation, 2

syntax conventions
- bold, 1
- braces ({}), 1
- brackets ([]), 1
- ellipsis (...), 2
- italic, 1

lowercase, 1
mutually exclusive choices, 1
normal font, 1
optional elements, 1
punctuation and symbols, 2
repeated elements, 2
required elements, 1
subparameters, 2
uppercase, 1
vertical bars (|), 1

U

uppercase, 1

V

V* commands, 49
value buffer
handling signs, 45
SQL null values and indicators, 45
vertical bars (|), 1

Y

Y* commands, 49