Enhancements - Versions 8.1.1 and 8.1.2

This chapter describes the enhancements that have been made to Adabas for versions 8.1.1 and 8.1.2.

- Logical Extent Limit Lifted
- Physical Extent Limit Lifted
- Expanded GCB
- MU and PE Limits Increased
- Expanded System File Numbers
- Spanned Record Support
- Large Object Field Support
- Direct Call Changes
- New Extended Adabas Control Block (ACBX)
- New Adabas Buffer Description (ABD)
- Buffer Enhancements
- FDT Enhancements
- Command Changes
- SVC Enhancements
- Utility Enhancements
- ADARUN Parameter Enhancements
- Command Log Timestamp Changes
- PRILOG Print Program Updates
- User Exit 11 and Sample Exit UEX11UX1
- Long Alpha (LA) Field Changes
- I/O Optimization
- Installation Updates
- New Licensing Component
- BS2000 Enhancements

- Hyperexit Support of Extended MU/PE Fields

Logical Extent Limit Lifted

The limit of five logical file extents for each Adabas file extent type has been lifted. The maximum number of logical file extents that can be defined is limited only in that the extent information of all address converters, Data Storage, normal index, and upper index extents for the file must fit into the file control block (FCB). (The extent information is stored in a variable section of the FCB.) For example, on a standard 3390 device type, a file could have more than 40 extents of each type (or there could be more of one type if there are less for another). For more information about Adabas 8 logical extents, read *Adabas Logical Extents*.

Physical Extent Limit Lifted

The Associator and Data Storage components of your Adabas database may now each contain more than five physical (database container) extents. A new maximum of 99 physical extents is now set for each. However, your actual real maximum could be less because the extent information of all Associator, Data Storage, and Data Storage Space Table (DSST) extents must fit into the general control blocks (GCBs). For example, on a standard 3390 device type, there could be more than 75 Associator, Data Storage, and DSST extents each (or there could be more of one extent type if there are less for another). For more information about Adabas 8 physical extents, read *Adabas Physical Extents*.

Expanded GCB

To accommodate all of the other expansions in Adabas 8, the general control block (GCB) now spans two blocks instead of one block. These blocks are now referred to as the *general control blocks (GCBs)*.

MU and PE Limits Increased

The number of occurrences of each MU field or each PE group in a record has been increased from 191 to about 65,534. However, the actual number of occurrences is limited to the size of the data-block size, the device type and the file type (spanned or not spanned). All MU fields and PE groups and other fields must fit into one compressed record. If you are using spanned records (introduced with Adabas 8), more MU fields and PE groups can be stored.

In addition, subdescriptors and superdescriptor definitions can affect the number of MU fields or PE groups in the record. For example, if a superdescriptor is created as a combination of a PE group and one or more MU fields and the number of occurrences is high, performance and resource problems can occur.

Note:
Excessive use of extended MU and PE fields might cause performance and resource problems. These can result in a work storage overflow, resulting in Response code 9. If this should happen, increase the ADARUN LP size for the database.

The use of more than 191 MU or PE fields in a record must be explicitly allowed for a file (it is not allowed by default). This is accomplished using the new ADADBS MUPEX function or the ADACMP COMPRESS MUPEX and MUPECOUNT parameters.

If a file has been established with extended MU or PE limits, you should not read the occurrence count of an MU field or PE group into a one-byte field in the record buffer. If you try, Adabas returns response code 55, subcode 9. Therefore, any application program that reads the occurrence count using an `xxC` element in the format buffer (for example, FB=’MUC.’ or FB=’MUC,1,B.’) must be changed to read the occurrence count into a field with two or more bytes (for example, FB=’MUC,2,B.’ or FB=’MUC,4,B.’).

- Identifying MU and PE Occurrences Greater Than 191

Identifying MU and PE Occurrences Greater Than 191

MU and PE occurrences greater than 191 are indicated in compressed records (ADACMP utility) by a `x’C0’` byte at the beginning of the occurrence count. This byte is set by the ADACMP utility or the nucleus when the records are compressed. The `x’C0’` indicator byte is followed by a byte indicating the number of count bytes used for the MU or PE occurrence count that follows. For example, consider the following indicator:

```
X’C0020204’
```

In this example, `x’C0’` indicates this is an extended count; `x’02’` indicates that there are two count bytes, and `x’0204’` indicates that there are 516 occurrences of the field.

Expanded System File Numbers

Two-byte file numbers can now be specified for all Adabas 8 system files (CHECKPOINT, SECURITY, SYSFILES, and TRIGGER), with numbers up to 5000. Physically coupled files, however, still cannot have numbers higher than 255. Note, however, that you cannot use the ADACNV utility to revert the database to a version prior to Adabas 8 once you implement two-byte system file numbers.

Spanned Record Support

Record spanning is implemented. The logical record may now be stored in multiple physical blocks. This expansion is provided to accommodate the extended features of Adabas 8, such as the increase in MU and PE occurrences. For more information, read *Spanned Record Support*.

Note:
Spanned record support must be explicitly allowed for a file. You can currently do this using the ADADBS RECORDSPANNING function or the SPAN parameter of ADACMP COMPRESS function.

To support spanned records, a secondary address converter is required. This address converter is used to map the ISNs of secondary spanned records to the RABNs of the Data Storage blocks where the secondary records are stored. For more information, read *Spanned Record ISN Use*.

Large Object Field Support

A new field option, LB, is provided to identify a field as a *large object field*. A large object field (LB field) is an alphanumeric field that can have a theoretical size of up to 2 GB. Such fields can be used, for example, to store documents (for example, HTML, XML, Microsoft Word, or PDF documents), pictures (for example JPG or BMP files), or other data conglomerates in single fields in the database.

Adabas stores LB field values in a separate file, called a *LOB file*, that is tightly associated with the file containing the LB fields, which is called the *base file*.

For more information about LB fields, read *Large Object Option LB* and *Getting Started with Large Object (LB) Fields*.

LB field support also includes utility enhancements. Specifically, enhancements to the ADACMP, ADADBS, and ADALOD utilities have been made. For more information, read about the *Utility Enhancements*.

Direct Call Changes

Adabas 8 introduces a new format of direct calls that is more powerful and flexible than the classic Adabas direct call interface of Adabas 7 and earlier:

- The data to be written to or read from the database in one command can be spread over multiple discontiguous buffer segments. It need not be put together into one physically contiguous buffer.
- Each buffer or buffer segment can be larger than 32 KB. For practical purposes, it can be as large as the application program’s working storage allows.

The new format of direct calls makes use of the extended Adabas control block (ACBX) and is called the ACBX direct call interface. The old format continues to use the classic Adabas control block (ACB) and is called the ACB direct call interface.

Adabas 8 understands both formats of direct calls. Your application programs can choose, on a call-by-call basis, whether to use the ACBX interface or the ACB interface. Your direct calls are only required to use the ACBX interface if the application needs to use one of the new features of the ACBX (multiple buffer segments, large buffers).

Adabas versions prior to Adabas 8 support only the ACB direct call interface. If they receive a call using the ACBX interface, a response code 22 will result (invalid command).

For more information, read *Calling Adabas*.

New Extended Adabas Control Block (ACBX)

The new extended Adabas control block (ACBX) introduced in Adabas 8 supports the ability to read and write data on the database using multiple, discontiguous, large buffer segments. This is especially useful for writing or reading large objects (LB fields), but can be used for any fields.

The existing ACB (non-extended) continues to be supported and your existing applications will continue to work, but if you want to take advantage of some of the extended features in Adabas 8, you must use the new ACBX.

For more information, read *Adabas Control Block Structures (ACB and ACBX)*.

New Adabas Buffer Description (ABD)

A new structure called an *Adabas buffer description* (ABD) is introduced with Adabas 8 and must be used in ACBX interface direct calls. ABDs cannot be used in ACB interface direct calls.

ABDs support the use of segmented, discontiguous buffers. Each ABD describes one buffer segment, identifying what kind of buffer segment it is, where it is located, what size it is, and other pertinent information.

For more information, read *Adabas Buffer Descriptions (ABDs)*.

Buffer Enhancements

With the introduction of the ACBX direct call interface, Adabas 8 offers various buffer enhancements for direct calls:

- New multifetch, performance, and user buffers can now be specified if you are making an ACBX interface direct call.
- Adabas format, record and multifetch buffers can be split into multiple segments, which need not be contiguous in storage.

  Format and record buffer segments are specified in pairs that belong together. For calls requiring multifetch processing, the format, record, and multifetch buffer segments are specified in triplets that belong together.
- More than 32 KB of data can be specified per buffer or buffer segment.

For more information, read *Defining Buffers*.

Furthermore, Adabas 8 offers format buffer enhancements for direct calls made using either the ACB or ACBX direct call interfaces:

- For large object (LB) fields, a zero length specified in the format element indicates that the amount of space available for the field value in the record buffer is variable and depends on the actual value of the field. The actual field value is determined at runtime and is stored in the first four bytes of the LB field value in the record buffer, preceding the field value itself. The length value is the sum of the length of the LB field value plus the four-byte length specification itself.
- For long alpha (LA) and LB fields only, an asterisk (*) can now be specified instead of a length in the format element. This indicates that the amount of space available for the field value in the record buffer is variable and depends on the actual value of the field. The actual field length is determined at runtime. However, unlike the zero length specification setting, *no* four-byte length field precedes the LA or LB field value in the record buffer; the record buffer area corresponding to the format element contains only the value of the field itself.

  For more information about these field length specifications, read *Length and Data Format* .
- A new format buffer indicator (*L*), referred to as the *length indicator*, can now be used to retrieve or specify the actual length of an LA or LB field value. This indicator is often combined with a format element with asterisk notation, where the latter specifies the field value proper and the former specifies the length of the value. For more information, read *Length Indicator (L)*.

- For LA and LB fields, the fixed length that can be specified in format elements has been extended from 253 bytes to a (theoretical) maximum of 2,147,483,647 bytes (2 GB).

For more information about these buffers, read *Defining Buffers*

FDT Enhancements

The internal FDT structure in version 8 has increased and these larger FDTs may use more than four Associator blocks. The additional blocks required for a larger FDT are automatically allocated from the Associator free space. The fixed space for FDTs in the Associator will remain reserved to accommodate backward compatibility and conversion.

Several new options have been added that you can use when defining fields in the FDT:

- A new LB (large object) option can be used to define large object fields.
- A new NB (no blank compression) option can be used in the definition of alphanumeric and wide-character fields in the FDT. Its presence indicates that trailing blanks in a field should not be removed by Adabas. By default, NB is not specified, and Adabas does remove trailing blanks when storing alphanumeric and wide-character values.

For more information about these FDT updates, read *Field Options*.

Command Changes

If an "I" is specified in Command Option 2 for an LF command, all field information is returned in the new Adabas 8 internal format.

In addition to the direct call changes, previously mentioned, LF command functionality has been altered to accommodate LB fields. If an LF command with Command Option 2 set to "S" is run and large object fields (LB fields) are encountered, the LB field description is returned in an F-type field element. Bit 6 in the second format byte (at offset 7 or byte 8 in the element) is set to indicate that the LB (large object) option is set for the field. In addition, bit 1 of the second format byte indicates whether the LB field is defined with the NB option. For more information, read *LF Command: Read Field Definitions* .

SVC Enhancements

Adabas 8 includes a new Adabas SVC. This SVC is fully backward compatible. In other words, you can use the new Adabas 8 SVC with Adabas 7 (or earlier) databases.

Note:
You cannot use the Adabas SVC from previous Adabas releases with Adabas 8 databases. If you attempt to do this, the Adabas 8 database will not initialize successfully.

On z/OS systems, the new SVC includes performance improvements and improved error recovery routines. Note that the new SVC shifts work from SRB-mode routines to TCB-mode routines. Take this information into account when analyzing Adabas 8 SVC performance. With the new SVC, SRB-mode overhead is largely eliminated and TCB-mode overhead is somewhat increased, but the net result is an overall improvement in SVC performance.

The new SVC on z/OS systems requires that the Adabas nucleus, as well as other MPM servers (such as Entire Net-Work and the Natural Global Buffer Pool), be APF-authorized. This APF authorization prevents use of the Adabas SVC by unauthorized programs that try to set themselves up as servers (which can receive and respond to calls from client programs). Software AG highly recommends that you run APF-authorized because of the security risks you can incur if you do not. However, upon request, Software AG does have a zap you can apply that eliminates this requirement. Contact your Software AG support representative for assistance.

Note:
Some add-on products require APF authorization to use restricted z/OS services. APF authorization is still required in these cases.

Utility Enhancements

In general, all Adabas utilities have been updated to support the new extended features in Adabas 8. Some of this support appears in the form of new or modified utility parameters. In other cases, support was added internally that does not affect your use of the utility at all.

The following table describes the Adabas utilities whose user interface has changed in this release. For complete information about the functions of any Adabas utility, read *Utilities*.

Utility	Summary of Changes
ADAACK	Spanned record support is provided in the ADAACK utility. However, ADAACK assumes that an ISN passed to it is a primary ISN or is the only ISN for a record. If an ISN is the primary ISN for a spanned record, all of the associated segment records of the spanned record are automatically checked in the secondary address converter. When printing error information about a particular ISN, the ADAACK utility will now indicate whether the problem is with a primary or secondary ISN, if the record is spanned.
ADACDC	Spanned records are not supported by the ADACDC utility at this time. However, when the IGNORESPANNED parameter is specified in an ADACDC run, ADACDC processing ignores spanned records, issues a warning message, and continues its processing. A return code of "4" is returned.

Utility	Summary of Changes
ADACMP	The length of the internal format buffer created by ADACMP from the FORMAT parameter for the COMPRESS or DECOMPRESS function is no longer limited to 64K bytes. The following *new* parameters have been added to the ADACMP COMPRESS utility to support the MU/PE extension, spanned records, and LB fields: • The DATADEVICE parameter specifies the Data Storage device type to be used for the segmentation of spanned records. If the SPAN parameter is specified, ADACMP will break long, spanned, compressed records into segments that are just a bit smaller than the Data Storage block size implied by the DATADEVICE parameter. If DATADEVICE is specified without the SPAN parameter, it is used to derive the maximum compressed record length that will be accepted. Longer compressed records will be considered in error and will be written to the DD/FEHL data set. • The HEADER parameter indicates whether or not the ADACMP compression logic should expect segmented ADACMP record headers in the uncompressed input records. The default is NO. (Contrast this with the HEADER parameter introduced in this release for ADACMP DECOMPRESS.) • The LOBDEVICE parameter identifies the device type that will be used for loading the *LOB file* produced by the COMPRESS function. • The LOBVALUES parameter indicates whether the uncompressed input data may contain long LB values (larger than 253 bytes). If LOBVALUES=YES is specified (allowed only for files with LB fields), ADACMP writes long large object (LB) field values to a second sequential output data set (identified by the DD/AUSB1 JCL control statement), which is to be loaded into the LOB file accompanying the base file loaded from the first output data set. • The MAXLOGRECLEN parameter can be used to specify the size, in bytes, of a buffer used by ADACMP to assemble the physical uncompressed segmented records into logical compressed records. This buffer is allocated and used only if HEADER=YES is also specified. • The MUPEX parameter indicates whether the extended MU/PE limits are allowed for the file. If this option is not specified, the maximum number of MU fields and the maximum number of PE fields that can be specified is 191. • The MUPECOUNT parameter specifies the size of the value count field in the input record for the COMPRESS function . Valid values are "1" or "2". If "1" is specified, each value count field preceding the MU or PE values in the input data must be one byte with a value of no more than "191". If "2" is specified, each value count field preceding the MU or PE values in the input data must be two bytes. A value count may exceed 191 only if the MUPEX parameter is also specified. • The SPAN parameter allows the record to be spanned after it is compressed if the compressed record exceeds the data storage block size of the device. The following *changes* have been made to existing ADACMP COMPRESS parameters: • The report produced by the DEVICE parameter now includes an indication of whether or not the MUPEX parameter has been set for a file. • The syntax of the FNDEF parameter has been changed, to allow you to specify the number of occurrences of the MU and PE options. You can also now specify the NB and LB field options. • The MAXPE191 parameter is no longer supported for the ADACMP COMPRESS function in Adabas 8. When specified, a warning message is issued and processing continues. • If USERISN is specified with HEADER=YES, the ISN immediately follows the ADAH header as part of the logical record. ADACMP DECOMPRESS processing supports the extended MU and PE limits and spanned records as input. The size of the value count preceding MU or PE values in each decompressed output record depends upon the extended MU and PE support for a file. A two-byte value count is given for files with extended MU and PE support. A one-byte value count is given for files without extended MU and PE support. In addition, the following functionality is supported. • The decompression of LB fields is supported. A new LOBVALUES parameter has been added. If the value of this parameter is set to "NO", only LB fields up to 253 bytes can be output from ADACMP DECOMPRESS processing. If the value of this parameter is set to YES, any LB field can be output from ADACMP DECOMPRESS processing. If "YES" is specified for the LOBVALUES parameter, the INFILE parameter must also be specified to identify the file number of the *base file* of a *LOB file group* whose data is to be decompressed. During processing, as ADACMP DECOMPRESS reads and decompresses the records from the base file, it reads any referenced LB field values from the *LOB file*. • A new HEADER parameter has been added to the ADACMP DECOMPRESS utility to indicate whether or not the ADACMP decompression logic should produce the ADACMP segmented record headers (ADAH and ADAC) as part of the decompressed output. The default is NO. • A new MAXLOGRECLEN parameter can be used to specify the size, in bytes, of a buffer used by ADACMP to assemble logical records that span one or more physical records of uncompressed output data. This buffer is allocated and used only if HEADER=YES is also specified. • The ISN parameter has been altered so that if it is specified with HEADER=YES, the ISN immediately follows the ADAH header as part of the logical record. Traditionally, the DD/FEHL error data set produced for ADACMP errors has truncated rejected records that exceeded the FEHL physical record length. In Version 8, the rejected records are segmented instead of truncated. Because of this change, the DD/FEHL LRECL setting must be at least 500 bytes.

Utility	Summary of Changes
ADADBS	Three new database services have been added to the ADADBS utility to support the extended MU/PE field counts and record spanning: • The MUPEX function allows you to specify the maximum number of occurrences allowed for MU or PE fields in a file. • The RECORDSPANNING function activates the use of spanned records in a file. • The SPANCOUNT function counts the spanned records in a file. In addition, large object (LB) fields can be added to the FDT using FNDEF definitions in the ADADBS NEWFIELD function. Finally, a new RESETPPT function resets the PPT blocks on the ASSO data set.
ADADCK	The ADADCK utility checks the header of a spanned record for plausibility, as follows: • The ISNs listed in the header are verified. The header contains the ISN of the primary record in the spanned record chain, the ISN of the previous spanned record in the chain, and the ISN of the next spanned record in the chain. • The spanned record identifier bits are checked. The header of a spanned record includes bits that indicate whether the record is a primary or secondary spanned record. Only one of these bits can be turned on in the header of any spanned record. A new MAXPISN parameter has also been introduced that allows you to set the maximum number of primary ISNs that will be checked for a spanned Data Storage file. The default is 1000.
ADAFRM	The ADAFRM utility can now be used to clear multiple PLOG headers from the PLOG, without requiring that you reformat the entire PLOG. To do this, you should specify the NUMBER parameter in conjunction with the FROMRABN parameter and set the SIZE parameter to "1". In addition, this utility can now handle the increased number of physical Associator and Data Storage extents (99) allowed in Adabas 8.
ADAICK	If you run the ADAICK DSCHECK function, the primary and secondary ISNs are now identified in the output.
ADALOD	The following new parameters are introduced for the ADALOD LOAD and UPDATE functions in support of spanned records and their associated secondary address converter: • The AC2RABN parameter allows you to specify or update space allocation for the secondary address converter. The secondary address converter is used to map the secondary ISNs of secondary spanned records to the RABNs of the Data Storage blocks where the secondary records are stored. • The optional MAXISN2 parameter allows you to specify the desired size of the secondary address converter (AC2) in ISNs. The secondary address converter is used to map secondary ISNs of secondary spanned records to the RABNs of the Data Storage blocks where the secondary records are stored. The following new parameters and parameter values are introduced for the ADALOD LOAD functions in support of large object (LB) fields and their associated *LOB files*: • The optional LOBFILE parameter allows you to specify the file number of the *LOB file* associated with a *base file*. This parameter is used when loading base files. • The optional BASEFILE parameter allows you to specify the file number of the *base file* associated with a *LOB file*. This parameter is used when loading LOB files. • A new file type, LOB, specified on the FILE parameter, allows you to indicate that you are loading an Adabas LOB file with a predefined FDT.
ADAORD	To support spanned records, the following two new parameters have been added to the REORASSO, REORDB, REORFASSO, REORFILE, and STORE functions of ADAORD. • The AC2RABN parameter allows you to specify the beginning RABN for the file's secondary address converter extent. • The optional MAXISN2 parameter allows you to specify the desired size of the secondary address converter (AC2) in ISNs. The secondary address converter is used to map the secondary ISNs of secondary spanned records to the RABNs of the Data Storage blocks where the secondary records are stored.

Utility	Summary of Changes
ADAREP	The report produced by ADAREP now lists whether the MUPEX and spanned record options have been set for the database. In the "Contents of Database" section of the report, the following changes have been made: • The padding factor has been removed from this table to make room for the larger extent values. However, it is still shown in the individual file details also provided in the report and it appears again if LAYOUT=1 is specified. • When LAYOUT=1 is specified during report creation, the larger extent values, supported by Adabas 8, appear for each file. • If a file is unable to build at least ten further file extents, ADAREP marks the file with an asterisk (*) on the right. In the "File Options" section of the report, a "T" indicates that two-byte MU/PE indexes are active for the file and an "S" indicates that use of spanned records has been activated for the file. In addition, the "Contains LOB Fields" column indicates whether the file contains one or more LB fields (an "L" appears if it does) and the "LOB File" column (the last column) indicates whether or not the file is a LOB file (an "L" appears if it is). Note that these two LB field columns are mutually exclusive; only one of them will be marked. In the "Physical Layout of the Database" section of the report, secondary address converter extents (for spanned records) are identified as "AC2" in the "Table File Type" column. In the "File Information" section of the report, a new field called "Two Byte MU/PE" indicates whether two-byte MU/PE indexes are active for the file. In the same section, the highest, maximum expected, and minimum secondary ISN are given as well as a new field called "Spanned Rec Supp", which indicates whether or not spanned records are activated for the file. In addition, the "Contain LOB Fields" field indicates whether or not the file contains one or more LB fields and the "LOB File" field indicates whether or not the file is a LOB file. In the Space Allocation section of the report, secondary address converter extents (for spanned records) are identified as "AC2" in the "List Type" column. Finally, three new checkpoints may be written by the Adabas nucleus: 75, 76, and 77.
ADASAV	The following new parameters are introduced for the ADASAV RESTONL FMOVE and ADASAV RESTORE FMOVE functions in support of spanned records and their associated secondary address converter: • The AC2RABN parameter allows you to specify the starting secondary address converter RABN for each file specified by FMOVE. • The MAXISN2 parameter allows you to specify the new number of secondary ISNs to be allocated for each file specified by FMOVE.
ADASEL	While there are no new parameters, ADASEL has been updated to support the MU/PE extension. In particular, when specifying indexes for MU or PE fields in an ADASEL SELECT IF statement, the indexes can now range from "1" through "65,534". Prior releases of Adabas restricted these indexes to values ranging from "1" through "191". ADASEL recognizes spanned records in its processing, but it cannot process files containing spanned records.
ADAULD	While there are no new parameters, ADAULD has been updated to support the MU/PE extension, LB fields, and and spanned records. In particular, two new statistics listing the number of record *segments* that were read and written during the run are produced by the ADAULD utility.

ADARUN Parameter Enhancements

The following ADARUN parameters have been added with Adabas 8:

- The CLOGLAYOUT parameter has a new valid value, 8, which supports the Adabas 8 command long format. For complete information, read *CLOGLAYOUT : Command Logging Format* .

  To support this new parameter, a new DSECT called LORECX that describes the CLOGLAYOUT=8 record layout is also provided. For additional information, read *Command Log Formats* .

- Three new LOGGING parameters have been added to support logging of Adabas 8 ABDs and multifetch and user buffers. These subparameters are LOGABDX, LOGMB, and LOGUB.

- A new CLOGBMAX parameter has been added. You can use this parameter to specify the maximum size of a logged buffer. If a buffer is longer, the logged buffer is truncated from the point at which its size exceeds the setting of the CLOGBMAX parameter. The CLOGBMAX setting affects the ADARUN LOGGING parameter specifications for both CLOGLAYOUT=5 and CLOGLAYOUT=8.

- A new CLOGMAX parameter has been added. You can use this parameter to specify the maximum size of all logged buffers used by an Adabas command. When the sum of sizes of the logged buffers for an Adabas command reaches the value of the CLOGMAX parameter, the buffer exceeding the limit is truncated and all following buffers are omitted. The CLOGMAX setting affects the

ADARUN LOGGING parameter specifications for both CLOGLAYOUT=5 and CLOGLAYOUT=8.

- Prior versions of Adabas tied the use of EXCP or EXCPVR for database I/Os to the APF-authorization of the load library. EXCP was always used when running non-APF-authorized; EXCPVR was always used when running APF-authorized. If you wanted to use EXCP when running APF-authorized, you were required to apply special A$- or AY- zaps.

  This release introduces a new ADARUN parameter, EXCPVR, available in z/OS environments. Using this parameter, you can specify whether EXCP or EXCPVR is used when running APF-authorized. For complete information on the use of this parameter, read *EXCPVR : Control EXCP or EXCPVR Use*.

  Finally, updates to the old A$- or AY-zaps will no longer be provided, as the zaps are no longer necessary.

- A new LNKGNAME parameter has been added. You can use this parameter to specify the name of the link globals table that should be used by an Adabas 8 batch/TSO link routine to obtain default information and to invoke any linked exits.

  Note:
  If you specify the LNKGNAME parameter, you must also specify the ADARUN DBID and SVC parameters.

- A new valid value, RENTUSER, has been added to the PROGRAM parameter. You can use this value to specify that a user program is to be run using a reentrant Adabas batch/TSO link routine.

- On BS2000 systems, a new parameter. SUBMPSZ. has been added that allows you to specify the common memory pool size in bytes for subtasks in products such as Adabas Review and Adabas Parallel Services.

For complete information about any ADARUN parameter, read *Adabas Initialization (ADARUN Statement)*.

Command Log Timestamp Changes

Timestamps in an Adabas 8 command log created using CLOGLAYOUT=8 are stored in machine time (GMT), whereas CLOGLAYOUT=5 timestamps are stored, as always, in local time. The LORECX record layout includes a differential time field that stores the difference between machine time and local time at the time the CLOG record is written. This field allows you to calculate the local time of a command log record. Because of the difference in timestamp formats, we do not recommend that you mix or merge command logs created using different CLOGLAYOUT settings (and, in fact, Adabas does not allow this in cluster or parallel services environments). For more information, read *CLOGLAYOUT : Command Logging Format* .

PRILOG Print Program Updates

The parameters for the PRILOG print program have changed:

- A new valid value, 8, has been added to the CLOGLAYOUT parameter. This value supports command logs created with ADARUN CLOGLAYOUT=8.
- New valid values (ACBX, MB, PB, and VERB) have been added to the FIELDS parameter. These valid values support the new and extended features of Adabas 8.
- A new parameter, DIMENSIONS has been added. This parameter allows you to specify the format of the printed output.
- New command selection parameters have been added that allow you to filter the PRILOG output.

For complete information about the PRILOG print program, read *PRILOG : Printing the Command Log* .

User Exit 11 and Sample Exit UEX11UX1

A new user exit 11 and the sample exit UEX11UX1 are supported in Adabas 8. Sample user exit UEX11UX1 can be used in front of your existing user exit 1 to get Adabas 8 to invoke the user exit 1 as user exit 11. This sample user exit can be used only if the direct call uses the ACB direct call interface, not the ACBX direct call interface. For more information, read *User Exit 1 (General Processing)*.

The new user exit 11 provides all the support of user exit 1, but for both ACB and ACBX-type direct calls. The only difference is that user exit 11 processes copies of the Adabas structures rather than the original structures themselves. For more information about user exit 11, read *User Exit 11 (General Processing)*.

Long Alpha (LA) Field Changes

Long alpha (LA) fields have been updated in Adabas 8 to include much of the support available for large object (LB) fields.

The following updates have been made to long alpha (LA) fields in this release:

- LA fields can now specify fixed field lengths greater than 253 in format buffers. For more information, read *Format Buffer Changes*.
- The new asterisk (*) field length specification is supported for LA fields in format buffers. For more information about the asterisk field length specification, read *Asterisk (*) Length Notation* .
- The new format buffer indicator (*L*), referred to as the *length indicator*, can now be used to retrieve or specify the actual length of an LA or LB field value. For more information, read *Length Indicator (L)*.
- For multiple value LA fields and LA fields within periodic groups, you cannot specify the base field without an occurrence index or with a "1-N" index. You must use a specific index or index range. For example, if L2 is an LA field with the MU option, the following format buffer specifications are *not* valid:

  ```
  FB='L21-N.'
  ```

  ```
  FB='L2.'
  ```

However, the following format buffer specification is valid, requesting the first three values of field L2:

```
FB='L21-3.'
```

I/O Optimization

In z/OS environments, various enhancements have been made to the Adabas 8 I/O routines, aimed at optimizing system resource use. Effective with this release of Adabas, Format-1 CCWs are used for I/O requests, whenever they are supported by z/OS (which includes all releases of z/OS with EXCPVR and z/OS 1.6 or later with EXCP). This assists in reducing the constraint on storage below the 16M line, since the I/O requests now occupy storage frames above the 16M line instead.

Wherever possible, Adabas I/O control blocks have been moved above the 16M line.

Finally, page-fixing requirements for each Adabas nucleus or utility using EXCPVR have been reduced because the amount of storage required to be fixed in Adabas 8.1.1 during each I/O request has been minimized. In addition, you can control how page-fixing occurs when EXCPVR is in use using the ADARUN PGFIX parameter introduced in Adabas 7.4.4. For complete information on the use of this parameter, read *PGFIX: EXCPVR Page Fixing Control*.

Installation Updates

While fully linked teleprocessing (TP) and batch monitor routines are provided with default values in Adabas 8, a new link globals module and associated LGBLSET macro have been introduced as well to tailor the installations of the TP and batch monitors under Adabas 8. This module and macro allow you to set the default values for the link routine components without editing Adabas 8 source and greatly simplifies the installation of the TP and batch monitors. In addition, various new modules have been added to support the Adabas 8 link routines.

To ensure that your existing Adabas 7 applications will still function in Adabas 8, the original Adabas 7 TP and batch modules are provided in Adabas 8, but with new names. You can continue to tailor these as you have in past releases.

Note:
All Adabas 8 CICS TP monitors use the task-related user exit (TRUE). The only non-TRUE installation you can perform is for Adabas 7 releases.

For more information about the TP and batch monitor installations, read *Installing Adabas With TP Monitors*.

New Licensing Component

A new licensing component is provided. This component cross-checks an assigned license key with your software installation and CPU IDs, and issues warning messages if inconsistencies are found. As a result of this new licensing component, nucleus startup changes are also necessary.

For complete information on licensing Adabas, read *Software AG Product Licensing*.

BS2000 Enhancements

The default device type for BS2000 in Adabas 8 is 2002. This device type is compatible with NK4 disks, whereas the old default device type of 2000 does not. In addition, the license file is assembled into a library which is in the nucleus BLSLIB chain.

Finally, a new ADARUN parameter was added for BS2000 environments: SUBMPSZ. This parameter allows you to specify the common memory pool size in bytes for subtasks in products such as Adabas Review and Adabas Parallel Services.

Hyperexit Support of Extended MU/PE Fields

Updated hyperexit logic is provided in this release to support the extended MU/PE fields in hyperdescriptor specifications.

In addition, Adabas 8 includes a Hyperexit Stub that allows your existing hyperexits to use the Adabas 8 parameter list without change. The Hyperexit Stub is intended as a temporary solution for those customers who do not wish to immediately update their hyperexits to use the new Adabas 8 parameter areas. For more information about all hyperexit support in Adabas 8, read *Hyperdescriptor Exits 01 - 31* .