Expanded Files

This chapter covers the following topics:

Overview

An expanded file is a logical file comprising one or more physical component files. Each component file contains records numbered by logical instead of physical ISN numbers. These physical component files must have

- *identical* field definition tables (FDTs); and
- *different* logical ISN ranges defined by the file's MINISN and MAXISN parameters. The ISN ranges cannot overlap.

The component files are chained together in sequence according to their ascending ISN ranges. The file with the lowest ISN range is called the "anchor" file; its file number is the number of the whole expanded file.

An expanded file can comprise up to 128 component files; it cannot exceed 4,294,967,294 records. An Adabas component file with 3-byte ISNs can contain a maximum of 16,777,215 records; a component file with 4-byte ISNs can contain 4,294,967,294 records.

Note:
Now that Adabas supports larger file sizes and a greater number of Adabas physical files and databases, the need for expanded files has, in most cases, been removed.

Expanded files are supported by the Adabas

- command that processes ISN lists, S8.
- sort commands S2 and S9. Before using this feature, investigate how it will affect database performance and impact users.

- prefetch/multifetch functions, which are enhanced for ET/BT support during expanded file processing.

Defining Expanded Files

Using ADALOD

Expanded files are defined at load time. Each physical component file is loaded separately using the ADALOD LOAD function. For all but the first component file, the ANCHOR parameter must be specified to refer to the anchor file. ADALOD LOAD then performs the following tasks:

- Compares the FDT of the new file to the FDT of the anchor file to ensure that they are the same.
- Checks the new ISN range (MINISN to MAXISN for the new component file) against the ISN ranges of the anchor and all other component files to ensure that there is no duplication;
- Checks for specification of the NOACEXTENSION parameter (Address Converter extensions are not permitted in component files);
- Checks that the MAXRECL parameter of the new component file is equal to that for the existing anchor file. All component files must have the same MAXRECL value.
- Loads the new component file, and;
- Links the new component file into the expanded file chain.

Example:

ADALOD LOAD statements define an expanded file (only the relevant parameters are shown):

```
ADALOD  LOAD FILE=11,NOACEXTENSION
ADALOD       MINISN=1,MAXISN=16000000
ADALOD       ...

ADALOD  LOAD FILE=23,NOACEXTENSION
ADALOD       ANCHOR=11
ADALOD       MINISN=36000001,MAXISN=50000000
ADALOD       ...

ADALOD  LOAD FILE=17,NOACEXTENSION
ADALOD       ANCHOR=11
ADALOD       MINISN=20000001,MAXISN=36000000
```

This example loads file 11 as an expanded file comprising:

File 11, ISN range:	1-16,000,000
File 17, ISN range:	20,000,001-36,000,000
File 23, ISN range:	36,000,001-50,000,000

Using the Online System

An expanded file can also be defined using the Adabas Online System *Define File* function. This function creates a new, empty file that can be specified as an anchor or component file for an expanded file. Existing files can be chained together using the *Expanded File Maintenance* function.

Rules for Defining Expanded Files

1. The NOACEXTENSION parameter must be set to prevent any extension of the Address Converter (i.e., increase to MAXISN) for the specified file.
2. The MINISN parameter must be specified when loading a component file for an expanded file.
3. The file number for the component file can be freely chosen.
4. A single file is loaded as an expanded file when the ANCHOR and FILE parameters specify the same file number.
5. An existing single file which is to be expanded may be referenced as the anchor file when the second component file is loaded. ADALOD then sets NOACEXTENSION for the first file, and makes it the anchor file.

   Note:
   An anchor file created in this way loses its anchor status when all component files are removed. If necessary, you can insert the file into itself to reestablish its anchor status.
6. The ISN ranges for the component files cannot overlap, but there may be gaps of unused ISNs between file ranges.
7. The component files can be loaded in any sequence.
8. If a new component file is loaded that has an ISN range lower than the range of the current anchor file, the newly loaded file becomes the new anchor file. The ANCHOR parameter of any component file loaded thereafter must refer to the new anchor file.

Inserting a Component File

Component files can be inserted into an expanded file using the ADALOD LOAD function as described in *Defining Expanded Files* or using Adabas Online System.

Using the online system, a new file can be created and inserted into an expanded file using the *Define File* function. A file that already exists can be inserted into an expanded file using the *Insert Component File* function.

Refer to the section *Rules for Defining Expanded Files* on page 57 for possible effects of adding a component file.

Removing a Component File

A component file may be both removed from the expanded file and deleted using the Adabas Online System *Delete File* function. To remove a component file from the expanded file chain without deleting the file, the Adabas Online System *Remove Component File* function can be used.

Refer to the section *Rules for Defining Expanded Files* on page 57 for possible effects of removing a component file.

Deleting Expanded Files

The Adabas Online System *Delete File* function also allows you to delete the complete expanded file; that is, to delete the anchor and all component files. The ADADBS utility's *DELETE* function can also be used to delete the complete expanded file.

Inspecting an Expanded File

In addition to the normal information about individual files, the report produced by the ADAREP utility shows the component file list for each expanded file in the database. The expanded file information itself is also available using the Adabas Online System *Display File* function.

Expanded Files and the Adabas Nucleus

A user call that refers to an expanded file is automatically directed to the appropriate physical component file by the Adabas nucleus. The user or application receives no indication that the selected file is an expanded file.

If the file number in the Adabas control block specifies the component file of an expanded file, the call is interpreted as being for the complete expanded file. Thus, user applications that accessed an existing component file in the past need not be changed if that file is integrated into an expanded file: the calls automatically apply to the complete expanded file. However, for convenience Software AG recommends that calls refer to the anchor file.

If a function performed on an expanded file produces results from more than one component file, those results are combined to produce a single result. For example, an L2 command (read physical sequential) for an expanded file is performed on each component file in sequence, beginning with the anchor file. Upon reaching the end-of-file for a component file, the L2 automatically continues with the next component file. The results are accumulated sequentially from all files that were read.

On the other hand, an L3 command (read logical sequential by descriptor) is performed as separate parallel calls to each component file, and the results are merged into a single sequence before they are returned to the caller.

Recommended Nucleus Changes for Expanded Files

To better accommodate parallel processing of component files for a single command, an increase in the following ADARUN parameter values is recommended for the nucleus session:

Parameter	Description
LI	Length of the table of ISNs (TBI)
LQ	Length of the table of sequential commands
LWP	Length of the Adabas work pool area
LS	Length of the search/sort area
NQCID	Maximum number of active command IDs (CIDs) allowed per user

Restrictions When Using Expanded Files

The following limitations apply to programs running on an expanded file:

- Physical and/or soft coupling is not currently supported for expanded files.
- Multiclient support is not provided for expanded files.
- Once established, component files of an expanded file cannot be renumbered.
- When Adabas Security is used for an expanded file, the following should be the same for all component files:
    - protection profile
    - password
    - security-by-value profile
    - cipher code

Expanded Files and Adabas Utilities

Although the expanded file is transparent to the user making Adabas calls, the DBA running the Adabas utilities must be aware of the existence of an expanded file. Adabas utility functions process expanded files in one of two ways:

- they process the complete expanded file; or
- they process component files.

Functions That Process Complete Expanded Files

Utility functions that process the entire expanded file include the following ADADBS, ADARES, and ADASAV functions:

ADADBS DELETE Function

Deletes a complete expanded file only.

ADARES REGENERATE and BACKOUT FILE Functions

Process the expanded file as a whole whenever one of the component files is specified in the file list. All other component files must then also be specified.

ADASAV

- **RESTORE(file)Function**
  Processes the expanded file as a whole whenever one of the component files is specified in the file list. All other component files must then also be specified.

- **SAVE (file)Function**
  Processes the expanded file as a whole whenever one of the component files is specified in the file list. When running the SAVE (file) function while the Adabas nucleus is active, all other component files must then also be specified.

Functions That Process Component Files

Utility functions that process component files include the following ADADBS, ADAINV, ADALOD, ADAORD, ADAACK, ADADCK, ADAICK, ADAVAL, ADAULD, and ADASCR functions.

ADAACK, ADADCK, ADAICK, ADAVAL, ADAULD

All functions of these utilities check single component files only.

ADADBS

- *CHANGE / NEWFIELD Functions.* These functions modify the field definition table (FDT) of a single component file only. The DBA must perform the CHANGE or NEWFIELD function for all component files in the expanded file. ADADBS prints a message indicating that the specified file is part of an expanded file, and then completes with condition code 4.

- *RELEASE Function.* Releases the index for a descriptor of a single component file. The DBA must perform the RELEASE function for all component files in the expanded file. ADADBS prints a message indicating that the specified file is part of an expanded file, and then completes with condition code 4.

ADAINV

- *INVERT Function.* Creates the index for a new descriptor of a single component file. The DBA must perform the INVERT function for all component files in the expanded file. ADAINV prints a message indicating that the specified file is part of an expanded file, and then completes with condition code 4.

- *COUPLE Function.* The ADAINV COUPLE function is not available for expanded files.

ADALOD UPDATE Function

Adds records to/deletes records from a single component file. When performing a mass update on some or all component files, the complete list of ISNs to be deleted from all component files can be supplied. ADALOD automatically selects only the ISN values from the specified range that are appropriate for the component file currently being processed. The same is true when adding new records with USERISN=YES.

When new records are being added with USERISN=NO but no free ISN is found, the loader cannot allocate a new Address Converter extent since the ISN range cannot be increased (NOACEXTENSION is active for all component files). Instead, ADALOD creates the index as though end-of-file had been reached. The remaining records not loaded may be added later to another component file using the SKIPREC parameter.

ADALOD does not check for unique descriptor values across component file boundaries.

ADAORD REORFILE / REORFASSO / REORDATA Functions

Each reorder the respective areas of a single component file. Since the file is not logically changed, the functions need not be performed on all component files of an expanded file.

ADASCR (Adabas Security) Functions

Defines security profiles for individual component files only. The protection, password, security-by-value and cipher code for each component file should be defined the same for all component files in an expanded file.