COUPLE: Define File-Coupling Descriptors

Use the COUPLE function to define one descriptor for each of two files to be coupled.

```
ADAINV   COUPLE    FILES = file-number1 , file-number2
                   DESCRIPTOR = 'fieldname , fieldname'
                   SORTSIZE = size
                   TEMPSIZE = size
                   [LPB = { prefetch-buffer-size | ADARUN-lu } ]
                   [LWP = { workpool-size | 1048576 } ]
                   [NOUSERABEND]
                   [PASSWORD = 'password' ]
                   [SORTDEV = {device-type | ADARUN-device } ]
                   [TEMPDEV = {device-type | ADARUN-device } ]
                   [TEST]
```

This chapter covers the following topics:

- Essential Parameters
- Optional Parameters
- Example
- Temporary Space for File Coupling
- Associator Coupling Lists
- Space for Coupling Lists
- Space Allocation

Essential Parameters

DESCRIPTOR: Descriptors Used as Basis for Coupling

The DESCRIPTOR parameter defines one descriptor in each file to provide the basis for coupling the files. Subdescriptors or superdescriptors may also be used, or may be defined as or derived from a multiple-value field. The descriptors specified may not be contained within a periodic group, nor be derived from a periodic group. The descriptors can have different names, but must have the same length and format definitions.

FILES: Files to Be Coupled

FILES specifies the two files to be coupled. The number of each file must be 255 or lower. The files specified may not be currently coupled to each other.

SORTSIZE: Sort Size

SORTSIZE specifies the space available for the sort data set or data sets R1/2 (SORTR2 is not supported under VSE). The value can be either cylinders (a numeric value only) or blocks (a numeric value followed by a "B"). If blocks are specified, they should be equivalent to a full number of cylinders. The SORTSIZE parameter must be specified. Refer to the *Adabas DBA Reference* documentation for more information on estimating the sort space.

TEMPSIZE: Temporary Storage Size

TEMPSIZE defines the space available for the temp data set. The value may be in cylinders (a numeric value only) or blocks (a numeric value followed by a "B"). This parameter must be specified.

Optional Parameters

LPB: Prefetch Buffer Size

LPB specifies the size, in bytes, of the internal prefetch buffer. The maximum value is 32760 bytes. The default depends on the ADARUN LU parameter; ADAINV may also reduce a specified LPB value if the LU value is too small.

LWP: Work Pool Size

LWP specifies the size of the work pool to be used for descriptor value sorting. The value can be specified in bytes or kilobytes followed by a "K". If no value is specified, the default is 1048576 bytes (or 1024K); however, to shorten ADAINV run time for files with very long descriptors or an unusually large number of descriptors, set LWP to a higher value. To avoid problems with the sort data set, a smaller LWP value should be specified when defining descriptors for relatively small files.

The minimum work pool size depends on the sort data set’s device type:

Sort Device	Minimum LWP	Minimum LWP
	Bytes	Kilobytes
2000	106496	104K
2314	090112	88K
3375	131072	128K
3380	139264	136K
3390	159744	156K

NOUSERABEND: Termination without Abend

When an error is encountered while the function is running, the utility prints an error message and terminates with user abend 34 (with a dump) or user abend 35 (without a dump).

If NOUSERABEND is specified, the utility will *not* abend after printing the error message. Instead, the message "utility TERMINATED DUE TO ERROR CONDITION" is displayed and the utility terminates with condition code 20.

PASSWORD: File Password

If one or both of the files being coupled is security protected, a valid password for the file (or files) must be specified with this parameter. If both files are password-protected, both must have the same password.

SORTDEV: Sort Device Type

ADAINV uses the sort data set to sort descriptor values. The SORTDEV parameter indicates the device type to be used for the sort data set. This parameter is required only if the device type to be used is different from that specified with the ADARUN DEVICE parameter. See the z/OS job control information for specific SORTDEV considerations.

TEMPDEV: Temporary Storage Device Type

ADAINV uses the temp data set to store intermediate data. The TEMPDEV parameter indicates the device type to be used for this data set. This parameter is required only if the device type to be used is different from that specified with the ADARUN DEVICE parameter.

TEST: Test Syntax

The TEST parameter tests the operation syntax without actually performing the operation. Only the syntax of the specified parameters can be tested; not the validity of values and variables.

Example

```
ADAINV COUPLE
FILES=3,4,DESCRIPTOR='AA,BB'
```

Files 3 and 4 are to be coupled. Descriptor AA from file 3 and descriptor BB from file 4 are to be used as the basis for the coupling.

Temporary Space for File Coupling

An intermediate data set is generated for *each* of the files being coupled.

An entry is written to the data set for each record contained in the file. Each entry contains the ISN of the record (3 or 4 bytes, depending on the ISNSIZE defined for the files) and the value (in compressed form) of the descriptor being used as the basis for the coupling. If the descriptor is defined with the NU option, no entries are written for records in which the descriptor contains a null value. If the descriptor is a multiple-value field, an entry is written for each different value.

The space required for *each* of the intermediate data sets is a function of the number of records contained in each Adabas file, and the length and the number of the different values present for the coupling descriptor in each record.

Use the following equation to determine the space needed for an intermediate data set:

```
SP = RECS x UV x (ISNSIZE + (AVLEN x 4)
```

where

SP	intermediate data set space required (in bytes).
RECS	number of records contained in the coupled file.
UV	average number of unique values per record for the descriptor. If the descriptor is not defined with the NU option, UV is equal to or less than 1. If the descriptor is defined with the NU option, UV is equal to the average number of values per record minus the percentage of records that contain a null value. For example, if the average number of values per record is 1 and 20 percent of the values are null, UV is equal to 1 - 0.2 = 0.8.
ISNSIZE	length of ISNs in the file (3 or 4 bytes).
AVLEN	average length (after compression) of each value for the descriptor.

Example: Calculating Intermediate Space Requirements for File Coupling

The file being coupled has 3-byte ISNs and contains 50,000 records. The descriptor being used as the basis for coupling contains 1 value per record (with no null values) and has an average value length of 5 bytes.

```
SP = 50,000 x  1 x  (3 + (5 + 1))
SP = 50,000 x  9
SP = 450,000 bytes
```

Associator Coupling Lists

ADAINV matches the two lists, sorts each resulting list, and writes each list to the Associator coupling lists.

The temp data set stores the matched (coupled) ISNs for each file. An entry is written to the temp data set for each match found. The entry contains the ISN of each record containing a matching value.

ADAINV sorts the entries stored on the temp data set using the sort area and writes the sorted entries to the Associator coupling lists for file A. The same process is then repeated for file B.

The temp area size requirement depends on the number of matching values in the two files for the descriptor used to couple the files. Each match requires 6 or 8 bytes, depending on the ISNSIZE defined for the files.

The sort area generally requires twice the amount of space as that needed for the temp area.

File coupling is bidirectional rather than hierarchical in that two coupling lists are created with each list containing the ISNs which are coupled to the other file.

Example: Coupling Lists

Assume that 2 files containing the descriptors AA and BB, respectively, are to be coupled. The values for the first five records of each file are as follows:

File A		File B	
ISN	**Field AA value**	**ISN**	**Field BB value**
1	20	1	18
2	25	2	40
3	27	3	25
4	30	4	20
5	40	5	20

If the two files were coupled using AA and BB as the basis for the coupling, the resulting coupling lists would be:

File A			File B		
ISN in FILE B*	**COUNT**	**COUPLED ISNs**	**ISN in FILE A***	**COUNT**	**COUPLED ISNs**
2	1	5	1	2	4,5
3	1	2	2	1	3
4	1	1	5	1	2
5	1	1			

** Internally, Adabas uses this field like a descriptor to determine the number and the ISNs of the coupled records.*

Space for Coupling Lists

The total space requirement for the coupling lists depends upon the number of common values that exist between the two descriptors used as the basis for the coupling.

The space requirement for *each common value* may be estimated as follows:

```
SP = 4a + 4b + 6ab
```

where

SP	space requirement for one common value (in bytes);
a	number of records in file A containing the common value;
b	number of records in file B containing the common value.

The total coupling list requirement is the sum of the space requirements of each common value.

Using sample files A and B as previously defined, space requirements per common value are

Common Value	Space Requirements
20	SP = 4(1) + 4(2) + 6(1 • 2) = 24 bytes
25	SP = 4(1) + 4(1) + 6(1 • 1) = 14 bytes
40	SP = 4(1) + 4(1) + 6(1 • 1) = 14 bytes

Total space required = 24 + 14 + 14 = 52 bytes

Example: Coupling List Space Requirements

Assume that 2 files are being coupled on the field ID. The values for ID are unique within each file. There are 5,000 common values in the coupled files.

Common Value	Space Requirements
n	SP = 4(1) + 4(1) + 6(1) SP = 14 bytes for one common value

There are 5,000 common values, each of which requires 14 bytes. The total space requirement for the coupling lists is 70,000 bytes.

Space Allocation

The coupling lists constructed by ADAINV are contained within the normal (NI) and upper (UI) index for each file being coupled. If the NI or UI component's logical extents currently allocated to the file are used up during ADAINV execution, ADAINV attempts to allocate an additional extent to the component. The size of the extent allocated is equal to 25 percent of the current total size of all logical extents currently assigned to the component. If insufficient space is available or if the maximum number of allocated extents has been reached for the component, ADAINV terminates with an error message.