Defining an Adabas Database

This chapter describes the procedure for defining an Adabas database. It is important for the DBA to understand the information provided for each step before attempting to define a new database.

Defining a new database involves the steps described in this chapter.

- Step 1 : Estimate the Size of the Database
- Step 2 : Allocate Space
- Step 3 : Format the Space
- Step 4 : Define Database Parameters

After you have completed these steps successfully, you can use the ADACMP and ADALOD utilities to load user files into the database.

- Step 1 : Estimate the Size of the Database
- Step 2 : Allocate Space
- Step 3 : Format the Space
- Step 4 : Define Database Parameters

Step 1 : Estimate the Size of the Database

This section covers the following topics:

- Components Required by the Nucleus
- Other Components
- General Space Requirements
- General Procedure for Estimating Space
- Estimation Formulas
- Normal Index (NI)
- Upper Index (UI)
- Address Converter (AC)
- Data Storage
- How Adabas Allocates Work Space

- Work Part 1: Data Protection Information
- Work Part 2: Intermediate Search Results
- Work Part 3: ISN Lists from Search Commands
- Work Part 4: Data Related to Distributed Transaction Processing
- Sort

Components Required by the Nucleus

The Adabas nucleus requires three database components: Data Storage, an Associator, and a Work area.

Data Storage

The Data Storage component contains the compressed data records of each file in the database.

Associator

The Associator contains elements for each file in the database and for the database as a whole.

For each file in the database, the Associator includes an inverted list, an address converter, and a field definition table (FDT):

- The *inverted list*, which resolves Adabas search commands and reads records in logical sequence, comprises the normal index (NI) and as many as 14 upper indexes (UI). All of the values for each descriptor in the file are contained in the NI along with a count of the records that contain each value and a list of the ISNs of those records. To increase search efficiency, UI levels are automatically created by Adabas as required, each level to manage the next lower level index. The first level UI, like the NI it manages, contains entries for only one descriptor in each index block. All other UI levels contain entries for all descriptors in each index block. Upper Indexes require a minimal amount of space (two blocks is the minimum).
- The *address converter* maps the logical identifier of a record (ISN) to the relative Adabas block number (RABN) of the Data Storage block where the record is stored. It comprises a list of RABNs in ISN order; for example, the fifteenth entry in the address converter contains the RABN for ISN 15.
- The *field definition table* (FDT) defines the logical contents of an Adabas file. It contains the name, level, length, format, and specified options for each field in the file.

For the database as a whole, the Associator includes storage management tables and coupling lists:

- *Storage management tables* list the Associator and Data Storage blocks that are available for allocation, along with the amount of unused space in each Data Storage block.
- *Coupling lists* exist for physically coupled files only are used to resolve search commands in which descriptors from more than one file are used.

Work

The Work area stores information in four parts:

- Part 1. Stores data protection information required by the routines for autorestart and autobackout.
- Part 2. Stores intermediate results (ISN lists) of search commands.
- Part 3. Stores final results (ISN lists) of search commands.
- Part 4. Stores data related to distributed transaction processing.

  Note:
  If you have Adabas Transaction Manager Version 7.5 or later installed, Work part 4 of DDWORKR1 is no longer supported. Instead, a second Work data set, DDWORKR4 is required. DDWORKR4 is a container data set used for the same purpose as Work part 4 of DDWORKR1, but it can be used in parallel by all members in a cluster. The DDWORKR4 data set should be allocated and formatted in the normal way, using a block size greater than or equal to DDWORKR1. It should be at least as large as the cluster's LP parameter of the database or cluster.

Other Components

Sort and Temp Areas

The Adabas utilities ADAINV, ADALOD, and ADAVAL require two additional data sets, SORT and TEMP, for sorting and intermediate storage of data.

The sizes of TEMP and SORT vary according to the utility function to be executed. These data sets can be allocated during the job and then released, or permanent data sets can be allocated and reused.

Logs

Adabas has the following optional logs:

- The *command log* (CLOG) records information from the control block of each Adabas command that is issued. The CLOG provides an audit trail and can be used for debugging and for monitoring usage of resources. Single, dual, or multiple (2-8) data sets can be used (multiple data sets are recommended).
- The *protection log* (PLOG) records before-images and after-images of records and other elements when changes are made to the database. It is used to recover the database (up to the last ET) after restart. Single, dual, or multiple (2-8) data sets can be used (multiple data sets are recommended).
- The *recovery log* (RLOG) records additional information that the Adabas Recovery Aid uses to construct a recovery job stream.

Note:
Each CLOG, PLOG, and RLOG data set is limited to 16,777,215 (x'FFFFFF') blocks/RABNs.

General Space Requirements

The space requirements for the Associator (NI, UI, and AC) and Data Storage are calculated automatically for each file by the ADALOD utility and the ADACMP utility, respectively. If you want to allocate a specific amount of space to a file or estimate the space needed for a file without actually executing these utilities, you can use the formulas provided in this chapter.

If the number and size of the files that will eventually be loaded into the database are not known at the time that the database is established, it is not necessary to allocate a large amount of extra space to the Associator and/or Data Storage, since the space may be increased subsequently by using the ADD or INCREASE function of the ADADBS utility.

The initial allocation for Associator and Data Storage should, however, allow for the loading of all currently planned files in addition to a reasonable amount of database expansion (adding new files or updating existing files).

When estimating the Associator space, the following requirements for the database as a whole must be added to the estimates calculated for each file within the database (normal index, upper indexes, and address converter):

- The first 30 Associator blocks are used by Adabas for storing internal control information. Note that the physical block sizes for Associator, Data Storage, and Work vary from one Adabas component to another and according to the device type on which each component is located.
- Associator blocks equaling five times the value specified by the MAXFILES parameter are reserved by Adabas for file control information. The MAXFILES parameter is set when running the ADADEF utility.

General Procedure for Estimating Space

1. Estimate the following requirements for *each file*; then add the estimates together for an estimate for the whole database:
   - Associator (normal index, upper indexes, address converter)
   - Data Storage
2. Estimate the following requirements for the *database as a whole*:
   - Associator (space reserved by Adabas)
     - First 30 blocks for internal control information;
     - (`MAXFILES * 5`) blocks for file control information (the ADADEF parameter MAXFILES specifies the maximum number of files that can be loaded into the database);
   - Work area; sort area; temp area; logs

Estimation Formulas

The following sections provide formulas for estimating the space that should be allocated to each component.

- Associator, in terms of
  - normal index (NI)
  - upper index (UI)
  - address converter (AC)
- Data Storage
- Work, in terms of
  - part 1 (data protection information)
  - part 2 (intermediate results of search commands)
  - part 3 (ISN lists from search commands)
  - part 4 (data related to two-phase commit processing)
- sort space

Rules of Precedence in the Formulas

The formulas follow the normal rules of precedence; that is, expressions are evaluated in the following order:

1. Elements in parentheses;
2. Multiplication and division operations;
3. Addition and subtraction operations;
4. Left to right (when elements have the same precedence level, the one on the left is evaluated first).

Normal Index (NI)

Use the following formula to estimate the normal index space required for each descriptor in the file:

```
NIRBYTES = ISNSIZE * AVUQVAL * RECORDS + DESCVALS * (AVLENG + 2)
```

where

NIRBYTES	is the space requirement for normal index, in bytes.
ISNSIZE	is the length of ISNs in the file (3 or 4 bytes). The ISN length is specified by the ADALOD parameter ISNSIZE.
AVUQVAL	is the average number of unique values for the descriptor in each record.
RECORDS	is the number of records to be contained in the file, which is specified by the ADALOD parameter MAXISN.
DESCVALS	is the number of unique values for the descriptor in the file.
AVLENG	is the average length of the values for the descriptor.

AVUQVAL

AVUQVAL is less than or equal to 1 unless the descriptor is a multiple-value field (MU) or part of a periodic group (PE).

If the descriptor is defined with the NU (null suppression) option, AVUQVAL equals the average number of values per record *minus* the percentage of records that contain a null value (the field is empty). For example, if each record has one value for the descriptor and 20 per cent of the values are null

```
AVUQVAL = 1 - 0.2 = 0.8
```

Similarly, if an MU field has an average of 10 values per record and 20% of the values are null

```
AVUQVAL = 10 - 2 = 8
```

AVLENG

If the descriptor field is not defined with the FI (fixed length) option, AVLENG equals the average compressed length of the field, including the length byte. If the descriptor is defined with the FI option, AVLENG equals the standard length of the field.

ISNSIZE * AVUQVAL * RECORDS

ISNSIZE * AVUQVAL * RECORDS represents the space required to store the ISNs. It is the important factor for descriptors that have many duplicate values.

DESCVALS * (AVLENG + 2)

DESCVALS * (AVLENG + 2) represents the space required to store the descriptor values. It is the important factor for descriptors that have a large proportion of unique values.

Convert Bytes to Blocks

Use the following formula to convert bytes to blocks. Round the result up to the next block.

```
NIRBLOCKS = NIRBYTES / (ASSOBLKSIZE * (1 - PADFACTOR / 100))
```

where

NIRBLOCKS	is the NI space requirement, in blocks.
NIRBYTES	is the NI space requirement, in bytes (from the NIRBYTES formula).
ASSOBLKSIZE	is the ASSOR1 block length. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).
PADFACTOR	is the ASSOR1 block padding factor, which is a percentage of the block length expressed as a value between 1-90.

Examples

The following examples assume that ASSOR1 is stored on a 3380 device.

Example 1:

Descriptor AA has one value per record and no null values. There are 50 different values for AA in the file. The average compressed length for the values is 3 bytes.

ISNSIZE=3
MAXISN=20000
PADFACTOR=10 (%)

```
NIRBYTES  = 3 * 1 * 20,000 + 50 * (3 + 2)
                    = 60,000 + 250
                    = 60,250 bytes

NIRBLOCKS  = 60,250 / (2004 * (1 - 0.1))
                       = 33.41
                       = 34 blocks
```

Example 2:

Descriptor BB has one value per record and no null values. There are 20,000 different values for BB in the file. The average compressed length for the values is 10 bytes.

ISNSIZE=4
MAXISN=20000
PADFACTOR=10 (%)

```
NIRBYTES  = 4 * 1 * 20,000 + 20,000 * (10 + 2)
                    = 80,000 + 240,000
                    = 320,000 bytes

NIRBLOCKS  = 320,000 / (2004 * (1 - 0.1))
                       = 177.42
                       = 178 blocks
```

Example 3:

Descriptor CC is a null-suppressed multiple-value (MU) field with an average of 10 occurrences and 3 null values per record. There are approximately 300 different values for CC in the file. The average compressed length for the values is 4 bytes.

ISNSIZE=3
MAXISN=10000
PADFACTOR=5 (%)

```
NIRBYTES  = 3 * 7 * 10,000 + 300 * (4 + 2)
                    = 210,000 + 1,800
                    = 211,800 bytes

NIRBLOCKS  = 211,800 / (2004 * (1 - 0.05))
                       = 111.25
                       = 112
```

Example 4:

Descriptor DD is a null-suppressed field contained within a periodic group. DD contains an average of 5 values per record; there is an average of 3 null values per record. There are 10 different values for DD in the file. The average compressed length for the values is 5 bytes.

ISNSIZE=4
MAXISN=10000
PADFACTOR=5 (%)

```
NIRBYTES  = 4 * 2 * 10,000 + 10 * (5 + 2)
                    = 80,000 + 70
                    = 80,070 bytes

NIRBLOCKS  = 80,070 / (2004 * (1 - 0.05))
                       = 42.06
                       = 43
```

Upper Index (UI)

Use the following formula to estimate the UI space required for each descriptor in the file:

```
UIRBYTES = NIRBLOCKS * (AVDESCLEN + ISNSIZE + RABNSIZE + 1)
```

where

UIRBYTES	is the UI space requirement, in bytes.
NIRBLOCKS	is the NI space requirement, in blocks (from the NIRBLOCKS formula).
AVDESCLEN	is the average compressed length of the values for the descriptor.
ISNSIZE	is the length of ISNs in the file (3 or 4 bytes). The ISN length is specified by the ADALOD parameter ISNSIZE.
RABNSIZE	is the length of RABNs in the database (3 or 4 bytes). The RABNSIZE is specified for all files in a database when the database is defined. **Note:** RABNSIZE refers only to the length of the relative Adabas block number. It does not refer to the block size.

Convert Bytes to Blocks

Use the following formula to convert bytes to blocks. Round the result up to the next block.

```
UIRBLOCKS = UIRBYTES / (ASSOBLKSIZE * (1 - PADFACTOR / 100))
```

where

UIRBLOCKS	is the UI space requirement, in blocks.
UIRBYTES	is the UI space requirement, in bytes (from the UIRBYTES formula).
ASSOBLKSIZE	is the ASSOR1 block length. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).
PADFACTOR	is the ASSOR1 block padding factor, which is a percentage of the block length expressed as a value between 1-90.

Example

This example assumes that the Associator is stored on a 3380 DASD; therefore, ASSOR1 has 2004 bytes per block.

The NI block requirement for this file is estimated to be 45 blocks. The average compressed length of the values for the descriptor is 3 bytes. The database has 3-byte (24-bit) RABNs; the file has 3-byte ISNs. The ASSOR1 block padding factor is 5 (%).

```
UIRBYTES  = 45 * (3 + 3 + 3 + 1)
                 = 450

UIRBLOCKS  = 450 / (2004 * (1 - 0.05))
                     = 450 / 1903.8
                     = 0.24
                     = 1 block( a minimum of 2 blocks can be allocated for the UI)
```

Address Converter (AC)

Use the following formula to estimate the address converter space required for the file. Round the result up to the next whole block.

```
ACBLOCKS = (MAXISN +!) * RABNSIZE / ASSOBLKSIZE
```

where

ACBLOCKS	is the space requirement for the address converter, in blocks.
MAXISN	is the MAXISN setting for the file.
RABNSIZE	is the length of RABNs in the database (3 or 4 bytes). The RABNSIZE is specified for all files in a database when the database is defined. **Note:** RABNSIZE refers only to the length of the relative Adabas block number. It does not refer to the block size.
ASSOBLKSIZE	is the Associator block size. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).

Examples

The following examples assume that the Associator is stored on a 3380 DASD; ASSOR1 has 2004 bytes per block.

Example 1:

MAXISN=2000000
RABNSIZE=3

```
ACBLOCKS  = (2,000,000 * 3) / 2004
                    = 6,000,000 / 2004
                    = 2994.01
                    = 2995 blocks
```

Example 2:

MAXISN=2000000
RABNSIZE=4

```
ACBLOCKS  = (2,000,000 * 4) / 2004
                    = 8,000,000 / 2004
                    = 3992.02
                    = 3993 blocks
```

Data Storage

Use the following formula to estimate the space required for Data Storage. Round the result up to the next whole block.

```
DATASTORAGE = MAXISN / ((DSBLKSIZE * (1 - (PADFACTOR / 100))) / AVRECLEN)
```

where

DATASTORAGE	is the space requirement for Data Storage, in blocks.
MAXISN	is the MAXISN setting for the file.
DSBLKSIZE	is the Data Storage block size, rounded down to the next integer. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).
PADFACTOR	is the Data Storage block padding factor, which is a percentage of the block length expressed as a value between 1-90.
AVRECLEN	is the average compressed record length.

Example

MAXISN = 1000000
Average compressed record length = 50
Model 3380 block size for DATA = 4820
Data Storage block padding factor = 5 (%)

```
DATASTORAGE  = 1,000,000 / ((4820 * (1 - 0.05)) / 50)
                         = 1,000,000 / (4579 / 50)
                         = 1,000,000 / 91
                         = 10,989.01
                         =  10,990 blocks
```

How Adabas Allocates Work Space

When you allocate the Work data sets, allocate enough space for all parts. The minimum allowable Work space is 300 blocks. Three ADARUN parameters can be used to break up the space into parts 1-4 as follows:

Notes:

1. You can use the DRES operator command to monitor your system’s use of Work parts 1, 2, and 3. For more information, read *DRES Command*.
2. Work part 4 or DDWORKR1 is no longer supported if you have Adabas Transaction Manager Version 7.5 or later installed. Instead, you should use the DDWORKR4 data set. DDWORKR4 is a container data set used for the same purpose as Work part 4, but it can be used in parallel by all members in a cluster. The DDWORKR4 data set should be allocated and formatted in the normal way, using a block size greater than or equal to DDWORKR1. It should be at least as large as the cluster’s LP parameter of the database or cluster.

- The ADARUN LP parameter specifies the size of Work part 1. The default setting is 1000 blocks; the minimum is 200. A database with little or no updating needs 500-1000 blocks. Work part 1 begins with RABN 1; the last RABN is the value of LP.

- The ADARUN LWKP2 parameter specifies the size of Work part 2. If LWKP2=0 (the default), Adabas calculates the size automatically, using the formula described in *Work Part 2: Intermediate Search Results* .

Work part 2 begins in the block following the last block of Work part 1; thus, the first RABN of part 2 is given by

```
1 + LP
```

- The ADARUN LDTP parameter specifies the size of Work part 4 when ADARUN DTP=RM. If LDTP=0 (the default), the length of Work part 4 is equivalent to the length of Work part 1 (ADARUN LP). If a non-zero value is specified, it must be greater than the value specified for LP. If a smaller value is specified, Adabas changes it to equal the LP value.

  Note:
  Work part 4 of DDWORKR1 is no longer supported if you have Adabas Transaction Manager Version 7.5 or later installed. Instead, you should use the DDWORKR4 container data set.

Work part 4 begins in the block following the last block of Work part 2; thus, the first RABN of part 4 is given by

```
1 + LP + LWKP2
```

- After allocating parts 1, 2, and possibly 4, Adabas allocates the remaining blocks to Work part 3. It is important that you allocate enough space to the DDWORKR1 data set to leave at least 50 blocks for part 3.

Work Part 1: Data Protection Information

The data protection area for all transactions running in parallel must fit into 1/4 of the Work part 1 (that is, LP) area. Following are general guidelines for determining the proper size for Work part 1:

1. The total Work part 1 size should be four times the estimated size required for a single average transaction in bytes times the maximum number of transactions that run in parallel. This value is then divided by the Work block size (in bytes) minus 200 to convert bytes to blocks.

2. If some transactions are very long, then those transactions alone plus all short transactions executed in parallel should be used to determine the size of a single average transaction.

3. The size of a single average transaction is determined by estimating the average number of updates (modifications, additions, and deletions) per transaction and multiplying that number by the estimated bytes required per update. To this is added space for ET data and for the ET record in bytes.

4. The size required per update is determined by the average compressed record length in bytes times 4 (before image, after image, and DVT space for each) plus 100 bytes for each protection record header (that is, 100 times 4).

A formula that expresses these guidelines is

```
WK1= ( 4 * TASIZE * TAP ) / ( BLKSIZE - 200 )
```

where

WK1	is size of Work part 1 in blocks
TASIZE	is the size of a single average transaction in bytes
TAP	is the maximum number of transactions actually executed in parallel
BLKSIZE	is the Work block size in bytes

```
TASIZE= ( ( ( 4 * AVCRL ) + 400 ) * UPDTA ) + ETDATA + 100
```

where

AVCRL	is the average compressed record length in bytes
UPDTA	is the average number of updates per transaction
ETDATA	is the average length of ET data in bytes

Example

If AVCRL = 300 bytes, UPDTA = 4, and ETDATA = 200 bytes, then

```
TASIZE= ( ( ( 4 * 300 ) + 400 ) * 4 )+ 200 + 100 = 6700 bytes
```

If TAP = 100 and BLKSIZE = 5492, then

```
WK1 = ( 4 * 6700 * 100 ) / ( 5492 - 200 ) = 506.46 blocks
```

Work Part 2: Intermediate Search Results

Use the following formula to estimate the space required for the Work part 2 area. Round the result up to the next whole block.

```
WORK2 = 22 + 2 * ((4 * RECORDS) / (BLKSIZE - 16))
```

where

WORK2	is the Work part 2 space requirement, in blocks.
RECORDS	is the number of records in the file with the most records. This number equals

```
TOPISN - MINISN + 1
```

where:

TOPISN	is the highest ISN currently used in the file.
MINISN	is the lowest ISN used in the file.

The MINISN value is specified with the ADACMP/ADALOD parameter MINISN; 1 is the default. You can use the ADAREP utility to display the TOPISN and MINISN values for the files in a database.

BLKSIZE	is the block size of the device where the Work data set is stored. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).

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

Example

The number of records in the largest file in the database is 500,000. The Work data set is stored on a 3380 device.

```
WORK2 = 22 + 2 * ((4 * 500,000) / (5492 - 16))
              = 752.46
              = 753 blocks
```

Work Part 3: ISN Lists from Search Commands

Adabas allocates to Work part 3 (resultant ISN lists) the Work space remaining after the allocation of the part 1 (data protection information) and part 2 (intermediate results) areas.

The *minimum* requirement for this area is 50 blocks.

If insufficient space is provided for this area, Adabas may be unable to execute additional search commands until the space currently occupied by ISN lists has been released. Consider the following factors when estimating the space needed for the Work part 3 area:

- The number of concurrent search commands to be processed (each ISN list is stored in a separate block), and the expected size of the resulting ISN lists (each ISN is stored as 4 bytes, regardless of the ISNSIZE specified for the file);

- The number of saved ISN lists resulting from previous search commands with the SAVE ISN LIST option which will be held concurrently;
- The amount of memory which will be required by Adabas as a result (each block allocated to this area requires 4 bytes of memory).

Example

A maximum of 100 search commands with an average of 25 resulting ISNs per command are to be processed concurrently during the session.

Adabas will need 100 blocks in the Work part 3 area.

Work Part 4: Data Related to Distributed Transaction Processing

Note:
Work part 4 of DDWORKR1 is no longer supported if you have Adabas Transaction Manager Version 7.5 or later installed. Instead, you should use the DDWORKR4 data set. DDWORKR4 is a container data set used for the same purpose as Work part 4, but it can be used in parallel by all members in a cluster. The DDWORKR4 data set should be allocated and formatted in the normal way, using a block size greater than or equal to DDWORKR1. It should be at least as large as the cluster's LP parameter of the database or cluster.

Work part 4 maintains information about some of the global transactions involved in distributed processing. For example, during phase one of the commit process, a global transaction's protection data may be copied from Work part 1 to Work part 4 if it is no longer possible to store the information in Work part 1.

If an overflow of Work part 4 is pending, the nucleus can force a transaction termination. This clears Work part 4 except for transaction IDs (XIDs) and local transaction status information.

The required size of Work part 4 depends on the applications running against the database and on the system load. If you have Adabas Transaction Manager Version 7.4 or earlier installed, a safe size of LP/4 is a good value to start with. If you have Adabas Transaction Manager 7.5 or later installed, a minimum size of 8 blocks must be specified, with the maximum number of blocks being the size of the DDWORK4 data set divided by 8.

Because the information maintained in Work part 4 cannot currently be moved to a different area, you can alter the size of Work part 4 between sessions only as follows:

- you can decrease the size of Work part 4 if it was not used at all in the previous session.
- you can increase the size of Work part 4 if it was used in the previous session.

Sort

The following formulas *estimate* the sort data set space used for sorting all values of a single descriptor. Multiple descriptors are sorted successively: all values are sorted for the first descriptor, then all values for the second descriptor, and so on. Therefore, estimate the space for the largest possible descriptor sort; that will be enough for all descriptors.

Use the following formula to estimate the space required for the sort area:

```
DESCSPACE = (AVDESCLEN + (1 + ISNSIZE)) * NUMRECS * AVPEOCCUR * AVMVOCCUR
```

where

DESCSPACE	is the total descriptor space required, in bytes.
AVDESCLEN	is the average compressed descriptor length, in bytes.
ISNSIZE	is the size of the ISN being used (either 3 or 4).
NUMRECS	is the number of records.
AVPEOCCUR	is the average number of periodic group occurrences, if the descriptor is in a periodic group. Otherwise, set this value to 1.
AVMVOCCUR	is the average number of multiple-value field occurrences, if the descriptor is a multiple-value field. Otherwise, set this value to 1.

Work Pool Size

Use the following formula to estimate the space required for the work pool:

```
LWPAVAIL = LWPSIZE - 1216 - (32 * SORTDEVTRKS) - SORTDEVBSIZ
```

where

LWPAVAIL	is the available part of the work pool space, in bytes.
LWPSIZE	is the total work pool size, in bytes (the utility's LWP parameter value).
SORTDEVTRKS	is the number of sort device tracks per cylinder. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).
SORTDEVBSIZ	is the sort device block size, in bytes.

Sorted Partial Sequences

To determine the space required for sorted partial sequences, use one of the following calculations. The one to use depends on the AVDESCLEN value (average descriptor length) used to calculate the DESCSPACE value (total descriptor space required).

- If AVDESCLEN is less than 12

```
LENGSEQ = (LWPAVAIL * (AVDESCLEN + (1 + ISNSIZE))) / 2
```

where

LENGSEQ	is the length of sorted partial sequences.
LWPAVAIL	is the available Work pool space.
AVDESCLEN	is the average compressed descriptor length, in bytes.
ISNSIZE	is the size of the ISN being used (either 3 or 4).

- If AVDESCLEN is equal to or greater than 12

```
LENGSEQ = (LWPAVAIL * 2) / 3
```

where

LENGSEQ	is the length of sorted partial sequences.
LWPAVAIL	is the available Work pool space.

Device Surfaces

Use the following formula to calculate the number of device surfaces rounded up to the next integer:

```
SURFACES = (DESCSPACE / LENGSEQ) / SORTDEVTRK
```

where

SURFACES	is the number of surfaces required for sort space, rounded up to the next integer.
DESCSPACE	is the total descriptor space required, in bytes.
LENGSEQ	is the length of sorted partial sequences.
SORTDEVTRKS	is the number of sort device tracks per cylinder. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).

Estimated Sort Size

Using the intermediate values calculated for LENGSEQ and SURFACES, compute the estimated sort size as follows:

```
SORTSIZE = (SURFACES * SORTDEVTRKS * LENGSEQ * 2) / (SORTDEVBSIZ - 4)
```

where

SORTSIZE	is the estimated sort area size, in blocks. This value should be rounded up to the next full cylinder.
SURFACES	is the number of surfaces required for sort space, calculated earlier and rounded up.
SORTDEVTRKS	is the number of sort device tracks per cylinder. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).
LENGSEQ	is the length of sorted partial sequences.
SORTDEVBSIZ	is the sort device block size, in bytes.

Number of Descriptors Sorted

Use the following formula to estimate the number of descriptors that can be sorted in the SORTSIZE space calculated in the previous formula (assuming the same descriptor definition that was used when calculating DESCSPACE):

```
DESCOUNT = SURFACES * SORTDEVTRKS * LENGSEQ / (AVDESCLEN + (1 + ISNSIZE))
```

where

DESCOUNT	is the number of descriptors defined in the earlier DESCSPACE calculation that can be held in the SORTSIZE space calculated above.
SURFACES	is the number of surfaces required for sort space, calculated earlier and rounded up.
SORTDEVTRKS	is the number of sort device tracks per cylinder. To review a list of block sizes by device type and component, refer to the sections entitled *Device And File Considerations* in the Adabas installation documentation for the appropriate system platform (z/OS, z/VSE, or BS2000).
LENGSEQ	is the length of sorted partial sequences.
AVDESCLEN	is the average compressed descriptor length, in bytes.
ISNSIZE	is the size of the ISN being used (either 3 or 4).

Step 2 : Allocate Space

1. Use standard operating-system procedures to allocate data sets for the following Adabas components:
   - Required by the Adabas nucleus:
     - Associator (ASSO)

    - Data Storage (DATA)
    - Work area (WORK1)
    - Work area (WORK4), if Adabas Transaction Manager 7.5 or later is installed.
  - Required by some Adabas utilities:
    - sort area (SORT)
    - temp area (TEMP)
  - Optional (but recommended) logs:
    - dual or multiple command log (CLOG)
    - dual or multiple protection log (PLOG)
    - recovery log (RLOG)

  Normally, ASSO, DATA, and WORK are each allocated as a single operating system data set. However, you can allocate the Associator and Data Storage on up to 99 separate data sets each; the data sets can be allocated on the same or different device types. Note that your actual real maximum number of physical extents may be less than 99 because the maximum number you can define is derived from the block size of the first Associator data set (DDASSOR1).

2. To minimize contention and distribute I/O activity more evenly across hardware channels, place the ASSO, DATA, WORK, PLOG, and RLOG data sets on different physical volumes. If only two volumes are available, place ASSO on one volume and DATA and WORK data sets on the second.

   The WORK and PLOG data sets should be on different volumes, since a PLOG I/O operation is always followed by a WORK I/O operation.

   The RLOG data set should always be placed on a separate device of the same type.

   Disk access time may be considerably reduced by separating TEMP from DATA, and SORT from ASSO. When loading files containing 100,000 records or more, splitting SORT across two volumes reduces disk arm movement.

3. Specify the disk space allocation in the job control (JCL/JCS) of the format utility (ADAFRM). See the Adabas Utilities documentation for specific information and job examples.

- Examples
- Performance Note

Examples

Example 1 : Database Allocation Using Two Volumes

Volume 1	Volume 2			
ASSO	DATA			
TEMP	WORK			
PLOG1	SORT			
PLOG2				

Example 2 : Database Allocation Using Three Volumes

Volume 1	Volume 2	Volume 3		
ASSO	DATA	WORK		
PLOG1	SORT	TEMP		
	PLOG2			

Example 3 : Database Allocation When Loading a Large File

Volume 1	Volume 2	Volume 3	Volume 4	Volume 5
ASSO	DATA	DATA	DATA	SORT (2nd half)
	PLOG1	PLOG2	SORT (1st Half)	WORK
		TEMP		

Performance Note

Software AG does not recommend using hardware compression (IDRC) for protection log files. The ADARES utility BACKOUT function will run at least twice as long under z/OS when processing compressed data. Also, the BACKOUT function is not supported for compressed data on z/VSE systems.

Step 3 : Format the Space

Before loading the first file into the database, use the ADAFRM utility to format the ASSO, DATA, and WORK data sets. Refer to the Adabas Utilities documentation for information about the ADAFRM utility.

Format TEMP and SORT before using any Adabas utility that requires them. You can allocate and format temporary data sets and delete them after executing the utility, or allocate and format permanent data sets for repeated use.

Note:
When using the Recovery Aid (including the RLOG), you must catalog all temporary data sets. When running with the Recovery Aid, the general rule is to catalog temporary data sets in jobs that require the Associator data sets.

Format the CLOG, PLOG, and RLOG data sets before starting the first session in which the logging is to be performed.

Step 4 : Define Database Parameters

Once all database components have been physically allocated and formatted, use the ADADEF utility to define database parameters such as database identification, maximum number of files, system file assignment, and so on.

The sizes of the ASSO, DATA, and WORK data sets must be defined with ADADEF DEFINE parameters. Note that defining the sizes to Adabas is different from allocating the space; the data sets must be allocated and formatted before you can define them to Adabas. The sizes of the other data sets are defined to Adabas as follows:

- TEMP and SORT: when you execute the utility that uses them;
- CLOG and PLOG: at the start of a nucleus session, with ADARUN parameters;
- RLOG: when logging begins, using the PREPARE function of the ADARAI utility.

Note:
Each log data set (CLOG, PLOG, or RLOG) is limited to 16,777,215 (x’FFFFFF’) blocks/RABNs.