# Device And File Considerations

This section provides information for the following device and system file topics:

- Supported Device Types
- ECKD Devices
- Adding New Devices
- Installing Adabas Using VSAM Data Sets

---

## Supported Device Types

The standard characteristics of the device types supported by Adabas on z/OS are summarized in the following table. Adabas block sizes and RABNs per track are provided for each Adabas component for each device type.

| Device | Trks/Cyl | ASSO | DATA | WORK | PLOG/RLOG | CLOG | TEMP/SORT/DSIM | Notes |
|---|---|---|---|---|---|---|---|---|
| 0512 | 16 | 2044:8 | 4092:4 | 8192:2 | 8192:2 | 8192:2 | 8192:2 | |
| 3310 | 11 | 2044:8 | 4092:4 | 4096:4 | 4096:4 | 4096:4 | 8192:2 | |
| 3330 | 19 | 1510:8 | 3140:4 | 4252:3 | 4252:3 | 3156:4 | 3140:4 | |
| 3340 | 12 | 1255:6 | 2678:3 | 3516:2 | 3516:2 | 3516:2 | 3500:2 | |
| 3350 | 30 | 1564:11 | 3008:6 | 4628:4 | 4628:4 | 3024:6 | 3008:6 | |
| 3370 | 12 | 2044:15 | 3068:10 | 5120:6 | 5120:6 | 3072:10 | 7680:4 | |
| 3375 | 12 | 2016:15 | 4092:8 | 4096:8 | 4096:8 | 4096:8 | 8608:4 | |
| 3380 | 15 | 2004:19 | 4820:9 | 5492:8 | 5492:8 | 4820:9 | 7476:6 | 3 |
| 3390 | 15 | 2544:18 | 5064:10 | 5724:9 | 5724:9 | 5064:10 | 8904:6 | 3 |
| 8345 | 15 | 4092:10 | 22780:2 | 22920:2 | 22920:2 | 22920:2 | 22920:2 | |
| 8350 | 30 | 3008:6 | 6232:3 | 9442:2 | 9442:2 | 9442:2 | 9442:2 | 1 |
| 8380 | 15 | 3476:12 | 6356:7 | 9076:5 | 9076:5 | 9076:5 | 9076:5 | 1 |
| 8381 | 15 | 3476:12 | 9076:5 | 11476:4 | 11476:4 | 9076:5 | 9076:5 | 1 |
| 8385 | 15 | 4092:10 | 23292:2 | 23468:2 | 23468:2 | 23468:2 | 23468:2 | 1 |
| 8390 | 15 | 3440:14 | 6518:8 | 10706:5 | 10706:5 | 8904:6 | 8904:6 | 1 |
| 8391 | 15 | 4136:12 | 10796:5 | 13682:4 | 13682:4 | 8904:6 | 18452:3 | 1 |
| 8392 | 15 | 4092:12 | 12796:4 | 18452:3 | 18452:3 | 18452:3 | 18452:3 | 1 |
| 8393 | 15 | 4092:12 | 27644:2 | 27990:2 | 27990:2 | 27990:2 | 27990:2 | 1 |
| 9332 | 6 | 2044:10 | 4092:5 | 5120:4 | 5120:4 | 10240:2 | 10240:2 | 2 |
| 9335 | 6 | 2556:14 | 3580:10 | 5120:7 | 5120:7 | 7168:5 | 7168:5 | |
| 9345 | 15 | 4092:10 | 7164:6 | 11148:4 | 11148:4 | 22920:2 | 22920:2 | 3 |

**Notes:**

1. The 8350, 838*n*, and 839*n* are pseudodevice types physically contained on a 3350, 3380, and 3390 device, respectively, but for which some or all of the standard block sizes are larger.
2. The number of tracks per cylinder listed here is artificial.
3. The IBM RAMAC 9394 emulates devices 3390 Model 3, 3380 Model K, or 9345 Model 2.

## Support for VSAM Data Sets

VSAM support is available only on z/OS.

To support VSAM data sets, the following table shows the CISZ values for the Adabas components on the 3380/90 devices:

| Device | ASSO | DATA | WORK | PLOG/RLOG | CLOG | TEMP/SORT/DSIM |
|---|---|---|---|---|---|---|
| 3380 | 2048 | 5120 | 5632 | 5632 | 5120 | 7680 |
| 3390 | 2560 | 5120 | 6144 | 6144 | 5120 | 9216 |

The VSAM device types 5555, 6666, 7777, 8888, and 9999 are dynamic device types that depend on the user definition.

A VSAM user can determine the RABN size currently in use from message ADAI64.

# ECKD Devices

Adabas supports ECKD DASD devices such as the IBM 3390 with the 3990 controller and ESCON channels.

During an open operation, ADAIOR determines which DASD device types are being used for the ASSO, DATA, WORK, SORT, and TEMP data sets. At that time, Adabas issues an informational message for each Adabas database component, where *type* is the component:

```
ADA164 ... FILE DDtype HAS BEEN OPENED IN ckd/eckd MODE - RABN SIZE rabn-size
```

**Note:**
Software AG strongly recommends that you avoid mixing ECKD and CKD extents within a file, because the file will be opened only in CKD mode. Mixing extents could degrade performance when file I/O operations are performed.

# Adding New Devices

Support for new device types that include user-defined block sizes can be implemented in ADAIOR by modifying one of the table of device-constant entries (TDCEs) reserved for this purpose.

A TDCE is X’40’ bytes long and the first free TDCE can be identified by X’0000’ in its first two bytes (TDCDT).

For all versions of Adabas prior to version 6.2, the address of the first TDCE is at offset ADAIOR+ X‘34’.

For Adabas Version 6.2, TDCE entries are in the ADAIOR CSECT TDCON: the first TDCE entry is at offset 0; the first free TDCE entry is at offset X‘400’.

For Adabas Version 7.1, TDCE entries are in the ADAIOS CSECT TDCON: the first TDCE entry is at offset 0; the first free TDCE entry is at offset X‘580’.

This information is valuable when adding an additional TDCE entry.

- Information to be Zapped into the First Free ADAIOR TDCE
- General Rules for Defining Device Block Sizes
- Maximum Sequential Block Size

- Rules for Associator and Data Storage Block Sizes
- Rule for Work Data Set Block Size
- Rules for TEMP/SORT Data Set Block Sizes
- Rules for PLOG or SIBA Block Sizes
- Sequential Protection Log Block Size in I_PPT

## Information to be Zapped into the First Free ADAIOR TDCE

The information in the following tables must be zapped into the first free TDCE. The rules described in the section *General Rules for Defining Device Block Sizes* must be followed when changing the TDCE.

| Label | Offset | Contents |
|---|---|---|
| TDCDT | 00 | Device type in unsigned decimal (X’3385’), must be numeric, and unique among all TDCEs. |
| TDCKSN | 02 | Constant set number: must be uniquely chosen from the values X’2B’ or X’2E’. |
| TDCF | 03 | The flag bit must be set—TDCFCKD (X’40’) for CKD devices, TDCFECKD (X’60’) for ECKD devices or TDCFECKD (X’61’) for ECKD, not user defined devices. |
| TDCDT1 | 04 | (see note 1) |
| TDCDT2 | 05 | (see note 1) |
| TDCDT3 | 06 | (see note 1) |
| TDCDT4 | 07 | (see note 1) |
| TDCMSBS | 08 | Refer to the section *Maximum Sequential Block Size*. |
| TDCTPC | 0A | Number of tracks per cylinder. |
| TDCCIPT | 0C | (see note 2) |
| TDCBPCI | 0E | (see note 2) |
| TDCABPT | 10 | Number of Associator blocks per track. |
| TDCABS | 12 | Associator block size. |
| TDCACPB | 14 | (see note 2) |
| TDCDBPT | 16 | Number of Data Storage blocks per track. |
| TDCDBS | 18 | Data Storage block size. |
| TDCDCPB | 1A | (see note 2) |
| TDCWBPT | 1C | Number of Work blocks per track. |
| TDCWBS | 1E | Work block size. |
| TDCWCPB | 20 | (see note 2) |

| Label | Offset | Contents |
|---|---|---|
| TDCTSBPT | 22 | Number of TEMP or SORT blocks per track |
| TDCTSBS | 24 | TEMP or SORT block size. |
| TDCTSCPB | 26 | (see note 2) |
| TDCPBPT | 28 | Number of PLOG blocks per track. |
| TDCPBS | 2A | PLOG block size. |
| TDCPCPB | 2C | (see note 2) |
| TDCCBPT | 2E | Number of CLOG blocks per track. |
| TDCCBS | 30 | CLOG block size. |
| TDCCCPB | 32 | (see note 2) |

**Notes:**

1. One or more operating-system-dependent codes for identifying the device type: z/OS, the UCB unit type from UCBTBYT4.
2. Not used for z/OS operating systems.

## General Rules for Defining Device Block Sizes

The following general rules must be followed when defining Adabas device block sizes:

- All block sizes must be multiples of 4.
- A single block cannot be split between tracks (that is, the block size must be less than or equal to the track size).

## Maximum Sequential Block Size

When adding new devices, the maximum sequential block size must also be specified. The value to be set to the maximum sequential block size is TDCMSBS, located at offset X’08’ from the beginning of the ADAIOR TDCE table.

Depending on the device type, the TDCMSBS value should be as follows:

| Device Type | Maximum Block Size |
|---|---|
| 0512 | 32760 |
| 3310 | 32760 |
| 3330 | 13030 |
| 3340 | 8368 |
| 3350 (8350) | 19069 |
| 3370 | 32760 |
| 3375 | 17600 |
| 3380 (8380/81) | 23476 |
| 339*n* | 27998 |
| 8380/1/5 | 23476 |
| 839*n* | 27998 |
| 9332 | 32760 |
| 9335 | 32760 |

**Note:**
On some devices, it may be most efficient to use smaller block sizes (for example, to specify 23476 for the 3380, but with two blocks per track).

## Rules for Associator and Data Storage Block Sizes

The following rules apply for Associator and Data Storage block sizes:

- Associator block size must be greater than one-fourth the size of the largest FDT, and should be large enough to accept definitions in the various administrative blocks (RABN 1 - 30) and in the FCB;
- The block sizes for Associator and Data Storage should be a multiple of 256, less four bytes (for example, 1020) to save Adabas buffer pool space.
- The Associator and Data Storage block sizes must be at least 32 less than the sequential block size.
- Data Storage block size must be greater than: (maximum compressed record length + 10 + padding bytes).

## Rule for Work Data Set Block Size

The Work block size must be greater than either (maximum compressed record length + 110) or (Associator block size + 110), whichever is greater.

## Rules for TEMP/SORT Data Set Block Sizes

If ADAM direct addressing is used:

```
size > (maximum compressed record length + ADAM record length + 24);
size > 277 (maximum descriptor length + 24)
```

However, TEMP and SORT are generally read and written sequentially; therefore, the larger the TEMP/SORT block size, the better.

Block sizes for TEMP and SORT must be greater than the block sizes for Data Storage.

## Rules for PLOG or SIBA Block Sizes

**Note:**
The use of 3480/3490 tape cartridge compression (IDRC) is not recommended for protection log files. The ADARES BACKOUT function will run at least twice as long under z/OS when processing compressed data.

The following rules apply for PLOG or SIBA block sizes:

- The PLOG or SIBA block size must be greater than either (maximum compressed record length + 110) or (Associator block size + 110), whichever is greater.
- It is also recommended that PLOG/SIBA be defined larger than the largest Data Storage block size. This avoids increased I/O caused by splitting Data Storage blocks during online ADASAV operations.

The block size (BLKSIZE) of a sequential file is determined as follows:

```
if PTTF(JCL) then BLKSIZE is taken from file assignment statement or label;
if PTTMBS > 0 then BLKSIZE = PTTMBS;
if PTTMBS = 0 then
if tape then BLKSIZE = 32760;
else BLKSIZE = TDCMSBS;
else if BLKSIZE in file assignment statement or label then use it;
if PTTF(OUT) then
if QBLKSIZE > 0 then BLKSIZE = QBLKSIZE;
if tape then BLKSIZE = 32760;
else BLKSIZE = TDCMSBS;
else error.
```

**Note:**
QBLKSIZE is an ADARUN parameter.

## Sequential Protection Log Block Size in I_PPT

In addition, the sequential protection log block size may have to be increased in the corresponding PTT entry in CSECT I_PTT of the load module ADAIOR.

The address of the first PTT entry is contained in the fullword at ADAIOR+X‘4C8’.

PTT entries begin at offset 0 into CSECT I_PTT.

Each PTT entry is X’10’ bytes long and has the structure given below:

| Label | Offset | Contents |
|---|---|---|
| PTTPN | 00 | Program number |
| PTTFT | 01 | File type |
| PTTN | 02 | DD name characters 2 - 8 |
| PTTF | 08 | Flags:<br>OUT (X’80’) output<br>BSAM (X’40’) BSAM<br>BACK (X’20’) read backwards<br>JCL (X’10’) BLKSIZE/LRECL/RECFM taken from DATADEF statement or label<br>UNDEF (X’04’) undefined record format<br>VAR (X’02’) variable record format |
| - | 09 | Reserved |
| PTTMBSZ | 0C | Maximum block size |

The PTT entry for the sequential protection log can be identified by X‘12F1’ in its first two bytes.

# Installing Adabas Using VSAM Data Sets

This section presents information needed to install Adabas on z/OS systems using VSAM as the access method for containing Adabas data. Software AG provides a VSAM interface as an alternative to using EXCP with BDAM container files.

- Suggested Additional VSAM Information Sources
- Defining VSAM Data Sets
- Defining Control Interval Sizes
- Using Existing Adabas Device Definitions
- Defining Your Own Device Characteristics
- Mixing VSAM and BDAM Components
- Converting Adabas BDAM Components to VSAM
- VSAM File Storage Requirements
- Allocating VSAM Data Sets on Multiple Volumes

- VSAM Limitations
- Comparison of EXCP and VSAM on High-Capacity Disk Drives

## Suggested Additional VSAM Information Sources

Software AG recommends that you have the following reference manuals when dealing with VSAM files. References are made in this section to these manuals:

- *IBM Access Method Services*
- *IBM VSAM Administration Guide, Macro References*

  See the list of manuals in the introduction for more specific information.

## Defining VSAM Data Sets

The following topics describe the VSAM file types used for Adabas files, and how to define and delete those files with the IDCAMS utility.

### VSAM File Types

There are four types of VSAM container files:

```
KSDS key sequential data sets
ESDS entry sequence data sets
RRDS relative record data sets
LDS linear data sets
```

Adabas uses RRDS or the optional LDS as the VSAM container file type that must be defined.

Normally, files are seen as containing records that can be read and written by the application program. Adabas uses the VSAM record to hold a block of compressed Adabas data. This means that the VSAM file contents is identical with that of its BDAM equivalent; only the access method is different. The RRDS and LDS VSAM file types were chosen because of their similarity to the current BDAM file structure.

### Defining a VSAM Data Set with the IDCAMS Utility

To define a VSAM data set, the IBM-supplied utility IDCAMS is used. This utility and its parameters are discussed in the manual *IBM Access Method Services*. It is not the intention here to describe IDCAMS in detail, but only those aspects that relate to Adabas requirements.

The following is an example job set-up for defining a VSAM data set for Adabas:

```
//IDC01 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
DEFINE CLUSTER ( NAME (EXAMPLE.ASSOR1) VOL (VOLXXX) -
CYLINDER(100) NUMBERED) -
DATA (NAME (EXAMPLE.ASSOR1.DATA) -
SHAREOPTIONS ( 3 3 ) -
CISZ (2048) -
RECORDSIZE (2004 2004) )
/*
//
```

The first SYSIN statement defines the VSAM cluster. The information that follows the SYSIN statement (within the outer parentheses) describe the VSAM file. The following is a short description of the parameters:

| Parameter | Description |
|---|---|
| NAME | Name assigned to the data set and used to refer to the data set in later jobs; for example, the following data definition refers to the name defined in the above example:<br><br>`//DDASSOR1 DD DSN=EXAMPLE.ASSOR1,DISP=SHR` |
| VOL | Volume or volumes on which this file is contained. |
| CYLINDER | DASD space required for this component. This may also be specified as RECORDS, TRACKS, etc. |
| NUMBERED | Identifies the file type as RRDS. |
| DATA (NAME) | Internal name describing the Data component of the VSAM file. |
| SHAREOPTIONS (3 3) | Defines how this data set is to be shared among other users. See the *IBM Access Method Services* manual for more information. |
| CISZ | Internal VSAM control interval size for this data set. |
| RECORDSIZE | VSAM record size; for Adabas use, this is the Adabas block size. |

The “-” character indicates continuation on the next job statement line.

## Deleting a VSAM Data Set

The IDCAMS utility is also used to delete a VSAM data set. The following is an example of a job for deleting a VSAM file:

```
//IDC01 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
DELETE (EXAMPLE.ASSOR1)
/*
//
```

## Multiple IDCAMS Operations

You can combine more than one IDCAMS utility operation. For example, you can delete and define a cluster or multiple clusters in one execution step. The following is an example:

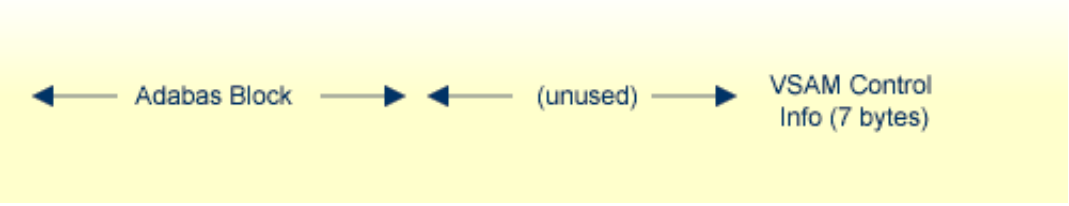
```
//IDC01 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
DEFINE CLUSTER ( NAME (EXAMPLE.ASSOR1) VOL (VOLXXX) -
CYLINDER(100) NUMBERED) -
DATA (NAME (EXAMPLE.ASSOR1.DATA) -
SHAREOPTIONS ( 3 3 ) -
CISZ (2048) -
RECORDSIZE (2004 2004) )
```

```
DELETE (EXAMPLE.DATAR1)
DEFINE CLUSTER ( NAME (EXAMPLE.DATAR1) VOL (VOLXXX) -
CYLINDER(100) NUMBERED) -
DATA (NAME (EXAMPLE.DATAR1.DATA) -
SHAREOPTIONS ( 3 3 ) -
CISZ (5120) -
RECORDSIZE (4820 4820) )
/*
//
```

## Defining Control Interval Sizes

A control interval is an area in which VSAM manages a record or group of records. VSAM maintains locks at the control-interval level; if an update is in progress for a record within a control interval, all records in that control interval are also locked, and cannot be accessed or changed. For this reason, only one record per control interval should be defined for Adabas VSAM files, since Adabas must be allowed to update or access any block on the database at any given time.

The following diagram shows a control interval containing an Adabas block:




How large the control interval is, depends on the record size. For records less than 8 KB, the control interval is defined in multiples of 512 bytes; for records equal to or larger than 8 KB, the control interval is defined in multiples of 2048 bytes.

To ensure that the correct CISZ is coded in your IDCAMS DEFINE request, use the following formula:

For records less than 8 KB:

```
resulta =  (recordsize + (7-byte CI overhead))  /  512
```

Round *resulta* up to the next higher number, then calculate CISZ:

```
CISZ = resulta (rounded)  *  512
```

For records equal or larger than 8 KB:

```
resulta =  (recordsize + (7-byte CI overhead))  /  2048
```

Round *resulta* up to the next higher number, then calculate CISZ:

```
CISZ = resulta (rounded)  *  2048
```

The following table shows the record sizes and CISZ values for the Adabas components on the 3380/90 devices:

| Device | ASSO | DATA | WORK | PLOG/RLOG | CLOG | TEMP/SORT |
|---|---|---|---|---|---|---|
| 3380 | 2004:2048 | 4820:5120 | 5492:5632 | 5492:5632 | 4820:5120 | 7476:7680 |
| 3390 | 2544:2560 | 5064:5120 | 5724:6144 | 5724:6144 | 5064:5120 | 8904:9216 |

## Using Existing Adabas Device Definitions

It is possible to use existing devices and device characteristics within the Adabas VSAM interface. No DEVICE-related changes are required in the ADARUN control parameters. The only requirement is that the IDCAMS RECORDSIZE parameter be identical to the block size of your existing BDAM component. However, it should be taken into account that this may lead to wasted disk space.

For example, a 3380 definition for the Associator requires a block size of 2004 bytes, plus seven bytes overhead per control interval and only one record per control interval. The actual use of the control interval is 2011 bytes, resulting in an unused area of 37 bytes (2048 - 2011 = 37). This unused space generally requires that the VSAM physical space allocation for the data set be larger than its BDAM equivalent.

It is not necessary that the VSAM data sets be on the same device type as that defined on the present ADARUN statements. For example, you may define the RECORDSIZE parameter for IDCAMS as a 3390 Adabas block size, while the VOL parameter points to a 3350 physical device type. In addition, VSAM allows the cluster to span different physical device types, while appearing to Adabas as a single device type.

## Defining Your Own Device Characteristics

When you define your own block sizes, the Adabas VSAM interface detects this and dynamically creates a device type of 9999 for the DD/*xxxx*R1 Adabas component. Or, if the VSAM interface detects an ADARUN DEVICE=9999 parameter, the VSAM file information is obtained from the VSAM catalog entry.

By defining your Adabas data sets with your own RECORDSIZE definition, you can ensure the best use of VSAM's control intervals. However, you must adhere to the guidelines for Adabas block sizes as defined in the section *General Rules for Defining Device Block Sizes*, except for the restriction that prohibits splitting a block between tracks. This can be done with the VSAM interface.

DD/*xxxx*R2 Adabas components (DD/PLOGR2, DD/ASSOR2, etc.) are defined using a dynamic device type of 8888, DD/*xxxx*R3 uses 7777, DD/*xxxx*R4 uses 6666, and DD/*xxxx*R5 uses 5555. For utilities that require DATADEV or ASSODEV, it is important to provide the appropriate dynamic device types according to this system.

The DD/*xxxx*R2-R5 data sets are only required for different Adabas block size definitions; they are no longer needed for defining VSAM files on different physical device types alone. A single component may span physically different devices in the VSAM interface, while still appearing to Adabas as being on the same device type.

## Mixing VSAM and BDAM Components

VSAM and BDAM components can be mixed. For example, you can have an Associator in a VSAM file with the rest of the Adabas components in BDAM files. But you cannot mix VSAM and BDAM within the same component. If, for example, you have DD/ASSOR1 and R2 defined, they must both be either

VSAM or BDAM files. Any Adabas component that currently resides on a BDAM file may be redefined on a VSAM data set. This includes Associator, Data, Work, PLOG, CLOG and the RLOG.

## Converting Adabas BDAM Components to VSAM

There are two ways to convert Adabas Associator (ASSO) and Data Storage (DATA) components to VSAM. The simplest method, described next, may require more DASD space for the VSAM components. The second method takes more time, but may save on DASD space.

### Method 1

**to convert ASSO and DATA components to VSAM (method 1):**

1. Run the ADAREP (database report) utility on the existing database with its BDAM components.
2. Run ADASAV SAVE on the database.
3. Run the IDCAMS utility to DEFINE the cluster or clusters for the Associator and/or Data Storage components being converted. Specify a RECORDSIZE that is consistent with these components, and a RECORDS value using the number of blocks indicated in the ADAREP output's "database physical layout" section for those components being converted. Remember to add one track's worth of blocks to the size reported there.
4. Following successful IDCAMS operation, run the ADAFRM utility on the VSAM files to format them. Remember to use the same device types as were defined on the BDAM database.
5. Run the ADASAV RESTORE function on the new VSAM data sets.
6. Convert all other nucleus and utility job control statements to apply to the VSAM data sets.

### Method 2

**to convert ASSO and DATA components to VSAM (method 2):**

1. Run the ADAORD RESTRUCTUREDB function on the BDAM-based database. Do not specify an Associator device type with ASSODEV different from the existing Adabas block size and device type definitions.
2. Run IDCAMS to allocate the VSAM files, using your own RECORDSIZE and size definitions. Define all Adabas Data Storage and/or Associator components.
3. Run the ADAFRM utility to format the VSAM files created in step 2. When using existing Adabas block sizes, use the existing device definition; otherwise, specify DEVICE=9999 to indicate dynamic device usage (see the section *Defining Your Own Device Characteristics* for more information).
4. Run the ADADEF DEFINE function on the VSAM files, and specify ASSOSIZE/DATASIZE according to the result of step 2, above, minus one track's worth of blocks.
5. Run the ADAORD STORE function on the VSAM files, and specify the correct device types. Use device type "9999" for dynamic device usage (see the section *Defining Your Own Device Characteristics* for more information).

Specify ADARUN TMLOG=NEVER for the purpose of verifying the installation. Once the verification process has been completed, reconsider this parameter setting.

6. Change all other nucleus and utility job control to specify the VSAM data sets defined and formatted in steps 2 through 5. Remember to change any device specifications to “9999” if you are using dynamic device definition (see the section *Defining Your Own Device Characteristics* for more information).

### Converting WORK, PLOG, CLOG and RLOG Components

**To convert WORK, PLOG, CLOG and RLOG components:**

1. Execute IDCAMS to define the cluster, using either the existing Adabas block sizes or your own RECORDSIZE definitions.
2. Execute ADAFRM to format the VSAM data sets.
3. Use the VSAM file or files in place of their BDAM counterparts in all Adabas nucleus and utility job control statements.

## VSAM File Storage Requirements

The Adabas VSAM interface makes use of BDAM user buffering and VSAM control interval processing to minimize VSAM overhead. However, this requires that the Adabas VSAM interface acquire storage to manage the contents of VSAM control intervals.

The Adabas VSAM interface uses up to 255 areas per Adabas file to manage VSAM control intervals. Each area is equivalent to the CISIZE specified in the IDCAMS definition for the file. These control interval areas are acquired dynamically while the nucleus or utility is executing; this minimizes the amount of storage required, based on the I/O response times of your environment. For example, in an environment where I/O performance is optimized, it is possible that only seven to ten of these CIAREAs would be needed to handle concurrent asynchronous VSAM requests.

In 31-bit addressing mode, these buffers are allocated above 16 MB. Software AG therefore recommends that you run with AMODE=31 when using the VSAM interface.

## Allocating VSAM Data Sets on Multiple Volumes

Allocating VSAM data sets across multiple volumes is done by specifying secondary space allocations as well as the volume serial numbers that will contain the VSAM data set. The following is a sample IDCAMS execution for defining EXAMPLE.ASSOR1 on volumes VOLXXX and VOLYYY with a primary space allocation of 100 cylinders and a secondary space allocation of 50 cylinders:

```
//IDC01 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
DEFINE CLUSTER ( NAME (EXAMPLE.ASSOR1) VOL (VOLXXX VOLYYY) -
CYLINDER(100 50) NUMBERED) -
DATA (NAME (EXAMPLE.ASSOR1.DATA) -
SHAREOPTIONS ( 3 3 ) -
CISZ (2048) -
RECORDSIZE (2004 2004) )
```

If the secondary allocation of 50 cylinders can be obtained on VOLXXX, it will be taken from that volume. Allocation of 50 cylinders will continue until VOLXXX can no longer satisfy an allocation. Then, space is acquired from VOLYYY.

The primary allocation must be acquired from VOLXXX; otherwise, IDCAMS will fail. The maximum number of VSAM extents is 123. On large databases, you must be careful to avoid fragmented VSAM file allocations that could cause the limit of 123 extents to be exceeded. For extremely large databases or for those requiring more than 123 extents, consider allocating additional database components (DD/*xxxx*R2-R5).

## VSAM Limitations

VSAM files can be defined to include up to 123 extents, or a maximum size of 4,294,967,296 bytes (4 gigabytes). Therefore, Adabas is permitted a maximum of 20 gigabytes of Associator (ASSO) and 20 gigabytes of Data Storage (DATA) components through the use of DD/xxxxR1 - DD/xxxxR5 data sets, and up to 4 gigabytes for all other Adabas components.

## Comparison of EXCP and VSAM on High-Capacity Disk Drives

Newly developed DASD (disk) devices have capacities considerably larger than any previous devices. The IBM 3390 Model 9 is one example. Such devices contain more than 65,535 tracks per volume, which makes it impossible to allocate a complete volume as a single data set, regardless of the number of extents. The 65,535 track limit is imposed by the operating system for most access methods.

To make use of a whole volume for ASSO or DATA files, it becomes necessary to allocate more than one container data set per volume, each data set not exceeding 65,535 tracks. These data sets would then be assigned to separate DATAR*n* or ASSOR*n* DD statements in Adabas JCL procedures. Since DATA and ASSO can each have up to five containers, this technique allows up to 327,675 tracks total for each. On a 3390-type device, this can be more than 18 GB.

If larger DATA or ASSO files are required, it is necessary to allocate one or more container files on multiple volumes. The operating system permits a single data set to span up to 59 volumes, as long as the total number of tracks allocated on each volume does not exceed 65,535. This allows DATA and ASSO each to be as large as 19,332,825 tracks, which can be more than 1 TB on 3390-type devices.

When deciding between EXCP and VSAM for such a device, the factors that should be considered are

- performance;
- maximum capacity; and
- ease of maintenance.

VSAM permits easier handling and improved SMS integration, but at the cost of somewhat lower performance. Also, since there can only be one VSAM cluster per operating system file (i.e., ASSOR1, etc.), this limits the total Associator or Data Storage size to 20 gigabytes. The maximum size for the Work data set is four gigabytes.

The capacity of high-capacity devices themselves may also restrict the choice. Unlike a BDAM file, a single VSAM cluster cannot exceed four gigabytes. This means that a single VSAM cluster cannot reference the complete volume on such a device (for example, the 3390 Model 9 has a maximum capacity of 8.5 gigabytes).

**Notes:**

1. EXCP offers high volume and performance, but with the disadvantage of requiring more maintenance.
2. Certain high-capacity drives have a slower data transfer rate. The performance impact of the data transfer rate must be taken into consideration when choosing the access method to be used on such devices.