

**Adabas**

**Adabas Installation for z/VM**

Version 7.4.4

September 2009

---

Adabas

This document applies to Adabas Version 7.4.4 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © Software AG 1971-2009. All rights reserved.

The name Software AG, webMethods and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. Other company and product names mentioned herein may be trademarks of their respective owners.

## Table of Contents

1	Adabas Installation for z/VM .....	1
2	About This Document .....	3
3	Supported Environments .....	5
4	Installation Procedure .....	7
	Installation Checklist .....	9
	Installing Using the Online Procedure .....	9
	Installing Without the Online Procedure .....	12
	z/VM Operating System Environment .....	13
	Adabas File Support under z/VM .....	37
	Entering Operator Commands .....	35
	Setting Defaults for ADARUN .....	36
	Installing New SM Levels .....	54
	Applying Zaps in z/VM .....	37
	IUCV Security Options .....	38
	Considerations for Installing ADALNK/ADAUSER .....	39
5	Device and File Considerations .....	45
	FBA Devices .....	46
	Adding New Devices .....	47
	General Rules for Defining Device Block Sizes .....	50
6	Installing The AOS Demo Version .....	53
	AOS Demo Installation Procedure .....	54
	Installing AOS with Natural Security .....	54
7	Installing The Recovery Aid (ADARAI) .....	57
	ADARAI Installation Overview .....	58
	ADARAI Installation Procedure .....	58
8	Managing UES Support of VM Databases .....	61
	Adding UES Support to an Existing Database .....	62
	Verifying UES Support .....	64
9	Adabas Dump Formatting Tool (ADAFDP) .....	65
	ADAFDP Function .....	66
	ADAFDP Output .....	66
10	Online Installation Program (INSTPROD/INSTADA) .....	73
	Loading INSTADA .....	74
	Running the Maintenance Version of INSTADA .....	75
	The INSTADA Display Panels .....	75
11	DATADEF Information For VM/GCS .....	99
	DATADEF File Assignments .....	100
	DATADEF Error Codes .....	102
12	Translation Tables .....	105
	Adabas EBCDIC to ASCII and ASCII to EBCDIC .....	106
	Entire Net-Work EBCDIC to ASCII and ASCII to EBCDIC .....	107
13	Glossary of Installation-Related Terms .....	109
	Index .....	113



# 1 Adabas Installation for z/VM

---

This document is intended for those who plan or perform Adabas installation for the z/VM operating system.

- *About This Document*
- *Supported Environments*
- *Installation Procedure*
- *Device and File Considerations*
- *Installing the AOS Demo Version*
- *Installing the Recovery Aid (ADARAI)*
- *Managing UES Support of VM Databases*
- *Adabas Dump Formatting Tool (ADAFDP)*
- *Online Installation Program (INSTPROD/INSTADA)*
- *DATADEF Information For VM/GCS*
- *Translation Tables*
- *Glossary of Installation-Related Terms*

Notation *vrs* or *vr*: When used in this documentation, the notation *vrs* or *vr* stands for the relevant version, release, and system maintenance level numbers. For further information on product versions, see *Version* in the *Glossary*.



## 2 About This Document

---

This document provides information for installing and configuring Adabas Version 7.4 on the IBM z/VM operating system.

Operating system requirements are provided, as well as procedures for installing Adabas and for adding new I/O devices.

### **Other Documentation You May Need**

The following Software AG documentation is referred to in this document and may be needed when installing Adabas:

- *Adabas Release Notes*
- *Adabas Operations*
- *Adabas DBA Tasks*
- *Adabas Triggers and Stored Procedures*
- *Adabas Command Reference*
- *Adabas Messages and Codes*
- *Adabas Utilities*
- *Adabas Dynamic Caching*
- *Adabas Online System*
- *Adabas Delta Save Facility*
- *Adabas Parallel Services*
- *Adabas Security* (available only on written request from an authorized user site representative)

The following IBM manuals are also referred to in this manual and may be required for installing or maintaining Adabas under z/VM:

- *Planning and Administration (SC24-5521)*

- *Operator's Guide (SC24-5528)*
- *Installation Guide (SC24-5526)*

Notation *vrs* or *vr*: If used in the following document, the notation *vrs* or *vr* stands for the relevant version, release, and system maintenance level numbers. For further information on product versions, see *Version* in the *Glossary*.



# 3 Supported Environments

---

Before attempting to install Adabas, ensure that the host operating system is at the minimum required level.

Adabas Version 7.4 is available for z/VM Version 4 Releases 2, 3 and 4.

Software AG provides Adabas support for the operating system versions supported by their respective manufacturers. Generally, when an operating system provider stops supporting a version of an operating system, Software AG will stop supporting that operating system version.

Although it may be technically possible to run a new version of Adabas on an old operating system, Software AG cannot continue to support operating system versions that are no longer supported by the system's provider.

If you have questions about support, or if you plan to install Adabas on a release, version, or type of operating system not mentioned above, consult Adabas technical support to determine whether support is possible, and under what circumstances.

---

# 4 Installation Procedure

---

▪ Installation Checklist .....	9
▪ Installing Using the Online Procedure .....	9
▪ Installing Without the Online Procedure .....	12
▪ z/VM Operating System Environment .....	13
▪ Adabas File Support under z/VM .....	37
▪ Entering Operator Commands .....	35
▪ Setting Defaults for ADARUN .....	36
▪ Installing New SM Levels .....	54
▪ Applying Zaps in z/VM .....	37
▪ IUCV Security Options .....	38
▪ Considerations for Installing ADALNK/ADAUSER .....	39

This section describes the preparation and installation of Adabas on IBM z/VM systems.

To use the information in this section, you need to be familiar with z/VM and CP concepts such as directories and virtual machines. See the [Glossary](#) for an explanation of terms referenced.



**Note:** See section [VM/GCS](#) for VM/GCS-related information to support some Software AG products such as Entire Net-Work. Adabas itself no longer supports VM/GCS.

## Installation Checklist

The following list provides an overview of the Adabas installation procedure on a z/VM system.

Step	Description	Additional Information
1	Define virtual machines for the Adabas environment.	Two virtual machines are required for operating Adabas. At least two more virtual machines are recommended: <ul style="list-style-type: none"> <li>■ Adabas nucleus (required)</li> <li>■ ID table manager (required)</li> <li>■ database administrator (recommended)</li> <li>■ user programs (recommended)</li> </ul> For specific information on the CP directory requirements for each of these virtual machines, refer to the individual sections describing the virtual machines.
2	Define minidisk space for the Adabas library disk.	If a DBA virtual machine has been defined, the library minidisk should be defined by an MDISK statement in the CP directory for the DBA virtual machine; otherwise, it should be defined in the CP directory of the Adabas nucleus machine. For detailed information about the Adabas library minidisk see the section <a href="#">Adabas Library Requirements</a> .
3	Define minidisk space for the Adabas database.	Some effort should be made to place these minidisks on different channels, both virtual and real. For detailed information about the Adabas database minidisks, see the section <a href="#">Disk Space Requirements for the Database</a> .
4	Load the Adabas release tape and subsequently install a database.	This can be done using either the online procedure (INSTPROD) as described in the following section and in section <a href="#">Using INSTPROD (Online Install)</a> , or manually; however, the online procedure is recommended.



**Note:** If you plan to use the Adabas Recovery Aid (ADARAI) utility, specific changes must first be made to the PROFILE and ADF<sub>dbid</sub> EXECs. Neither INSTADA nor the ADAMAINT EXEC are currently capable of making the changes needed for running ADARAI. For more information, see the section [VM/ESA or z/VM](#) in the ADARAI utility documentation.

---

## Installing Using the Online Procedure

---

The following steps are required to load the release tape and install Adabas using the online procedure:

▶ **to load the release tape and install Adabas using the online procedure:**

- 1 LOGON to and then DISCONNECT from the ID table manager virtual machine.
- 2 LOGON to the Adabas nucleus virtual machine. If a database administrator (DBA) virtual machine has been defined, DISCONNECT from Adabas; otherwise, ATTACH a tape drive to Adabas as drive 181 and ignore step 3.
- 3 LOGON to the DBA virtual machine and ATTACH a tape drive as drive 181.
- 4 ACCESS the Adabas library disk with a filemode other than "A". Filemode "C" is recommended.
- 5 Issue command `TAPE FSF 1.`
- 6 Issue command `TAPE LOAD * * filemode`, where *filemode* is the mode specified in step 4.
- 7 Issue INSTPROD to start the online installation procedure (after the files are unloaded from the installation tape, INSTPROD starts INSTADA automatically).
- 8 Enter the installation parameters on the online screens that are presented. Section [Using INSTPROD \(Online Install\)](#) contains a detailed description of the INSTADA procedure.
- 9 Customize the ADAINPL EXEC file for your environment as preparation for loading the Adabas Online System (if used) into a Natural system file.
- 10 Stop the Adabas nucleus by entering ADAEND from the console of the Adabas virtual machine, or from a secondary console.
- 11 Execute the ADASAV EXEC, specifying the filename of the z/VM file that contains the ADASAV control statements (i.e., SAVE) and the DBID as parameters.

This completes the installation using the online procedure. If the installation is interrupted, INSTADA can be restarted; all the parameter values already entered are stored.

Existing installations may use the INSTADA procedure to convert an Adabas 6 database to Adabas 7. In this case, the `CONVERT` parameter should be set to YES.

The EXECs supplied on the Adabas release tape are based on the control statement files and EXECs created by INSTADA. Software AG strongly recommends that new databases be defined and formatted using only the INSTADA EXEC from step 7.

Additional databases can be defined using the INSTADA EXEC. The parameters for each database are entered in the online procedure panels during the initial installation. Section [Using INSTPROD](#)

*(Online Install)* describes how INSTADA can be used, and shows the INSTADA screens and menus.

The ADAMAINT EXEC can be used to maintain the Adabas environment, including updating all files created by INSTADA. This EXEC presents the same panels as INSTADA, but does not format the database. This prevents accidental destruction of a database while performing maintenance. The ADAMAINT EXEC allows you to either increase a dataset or add a new extent by making changes to the online screen.

The INCREASE EXEC can be used to run the INCREASE or ADD functions of the Adabas ADADBS utility as well as formatting with ADAFRM. The ADAMAINT EXEC must be run before using the INCREASE EXEC.

## Installing Without the Online Procedure

---

The following steps are required to load the release tape and install Adabas without the online procedure. Many of the steps correspond to steps performed automatically by the online procedure, but not necessarily in the same order.

▶ **to load the release tape and install Adabas without using the online procedure:**

- 1 LOGON to and DISCONNECT from the ID table manager virtual machine.
- 2 LOGON to the Adabas nucleus virtual machine. If a database administrator (DBA) virtual machine has been defined, DISCONNECT from Adabas; otherwise, ATTACH a tape drive to Adabas as unit 181 and continue with step 4.
- 3 LOGON to the DBA virtual machine and ATTACH a tape drive as drive 181.
- 4 ACCESS the Adabas library disk with a filemode other than "A". Filemode "C" is recommended.
- 5 Issue command `TAPE FSF 4`.
- 6 Issue command `TAPE LOAD * * fm`.
- 7 Create a PROFILE EXEC for the DBA virtual machine.

Refer to the section *z/VM Operating System Environment* for an example of the CP directory entries for the DBA virtual machine.

Include the following functions in the PROFILE EXEC:

- MULTI write LINK commands to the Adabas database minidisks (ASSO, DATA, WORK, TEMP, SORT).
- ACCESS for the library disk as a read only extension of the A-disk (`ACC cuu fm/A`).
- Issue EXEC SETTXTLB.

- Define ADARUN, DATADEF, DISPDD, and RELDD as nucleus extensions. This can be done by executing the NUCXTNTS EXEC with no parameters.
- 8 Execute the PROFILE EXEC.
  - 9 Create a zap file for ADAITM if you are going to modify the default values for the target-ID, VMID, or restart parameters:
    - target ID of ADAITM at location X'20'. (default X'FFFF').
    - VMID of the DBA virtual machine at location X'28'. (default: 8 blanks)
    - automatic restart parameter at location X'30' (default: C'Y').

- 10 Create a zap file for ADALDI if you are going to modify the default values for the VMID of the ID table manager.

The VMID of the ID table manager is at location X'34' (default C'DBIDSERV').

Apply the zaps if necessary.

- 11 Issue the following commands, where *nnnnn* is the database ID and *fm* is the filemode. Modify the copied EXECs, which always have a filename containing the five-digit DBID (with leading zeros, if needed), as follows:

```
COPY ADADEFS EXEC fm ADFnnnnn EXEC fm
COPY ADAFRM CONTROL fm FRMnnnnn CONTROL fm
COPY ADADEF CONTROL fm DEFnnnnn CONTROL fm
COPY RUNDB CONTROL fm RDBnnnnn CONTROL fm
COPY RUNDEV CONTROL fm RDVnnnnn CONTROL fm
```

- 12 Copy the ADALOD control statements to load the demo files:

```
COPY loadname CONTROL fm demoname LODLIB fm
```

—where *loadname* is EMPLLOD, VEHILOD or MISCLOD, and *demoname* is EMPL, VEHI, and MISC, respectively. Change the copied files as required.

- 13 Create the file DBnnnnn VOLUMES *filemode*, where *nnnnn* is the five-digit database ID.

This file must contain one record per direct access mini-disk (that is, one for the Associator, one for Data Storage, and so on). Each record must contain the file name, file type, disk label, and virtual unit address (in that order) where the disk will be formatted and reserved. The record entries must be separated by blanks.

Copy the ADALOD control statements to load the demo files.

- 14 Execute the DBINIT EXEC specifying the DBID as a parameter.

- 15 Execute the ADALOD EXEC specifying FILENAME (EMPL,VEHI,MISC) and the DBID as parameters.
- 16 LOGON to the ID table manager virtual machine.
- 17 Create a PROFILE EXEC for the ID table manager virtual machine.

Refer to the section *z/VM Operating System Environment* for an example of the CP directory entries for the ID table manager virtual machine.

Include the following functions in the PROFILE EXEC:

- multi-read LINK commands to the Adabas library disk (ADAV<sub>vvv</sub> TXTLIB).
- ACCESS for the library disk as a read only extension of the A-disk (ACC *cuu fm/A*);
- EXEC SETTXTLB;
- LOAD ADAITM \* (START.

- 18 Execute the PROFILE EXEC created in step 17.
- 19 DISCONNECT from the ID table manager virtual machine.
- 20 LOGON to the Adabas virtual machine.
- 21 Create a PROFILE EXEC for the Adabas virtual machine.

Refer to the section *z/VM Operating System Environment* for an example of the CP directory entries for the Adabas virtual machine.

Include the following functions in the PROFILE EXEC:

- multi-read LINK commands to the Adabas library disk (ADAV<sub>vvv</sub> TXTLIB).
- ACCESS for the library disk as a read-only extension of the A-disk (ACC *cuu fm/A*).
- EXEC SETTXTLB
- \* SET STORECLR ENDCMD
- EXEC NUCXTNTS (defines ADARUN, DATADEF, DISPDD, and RELDD as nucleus extensions).

- 22 Execute the PROFILE EXEC created in step 21.
- 23 Start the Adabas nucleus by entering ADANUC, specifying the DBID as a parameter.
- 24 If a DBA virtual machine has been defined, DISCONNECT from the nucleus virtual machine before continuing; otherwise, continue with step 25.
- 25 LOGON to the database administrator (DBA) virtual machine.
- 26 Execute the ADAREP EXEC, specifying as parameters the DBID and the filename of the z/VM file containing the ADAREP control statements (i.e. REPCPLST).



- 27 Stop the Adabas nucleus by entering `ADAEND` from the console of the Adabas virtual machine, or from a secondary console.
- 28 Execute the `ADASAV EXEC` specifying the filename of the z/VM file that contains the `ADASAV` control statements (i.e. `SAVE`) and the `DBID` as parameters.

For systems using the Adabas Online System, perform the following additional step:

- 29 `LOGON` to the DBA virtual machine.

This completes the installation without the online procedure.

## z/VM Operating System Environment

---

In z/VM, the following entities are required or recommended to run Adabas.

- Adabas nucleus (required)
- ID table manager (required)
- database administrator (recommended)
- user programs (recommended)

Each of the above Adabas entities operates as a task within its own z/VM environment, known as a virtual machine. Each virtual machine comprises system resources that emulate virtual storage, virtual DASD (minidisk) space, and I/O capabilities such as an operator console and printer.

The actual system resource allocated to each Adabas virtual machine is either predefined in a CP directory or dynamically defined in the `PROFILE EXEC` that is invoked when each Adabas virtual machine begins operating. The following sections describe each of the Adabas virtual machines and how its resources are initially defined when installing Adabas.

Following this description is specific information concerning individual virtual machine requirements and operations in a z/VM environment.

### Adabas Nucleus Virtual Machine

The Adabas nucleus executes in its own virtual machine, which normally runs disconnected. The nucleus virtual machine requires various z/VM minidisks. The typical required virtual storage size for the Adabas nucleus machine is 6 MB.

- [Allocating Adabas Minidisk Space](#)
- [Communicating with Other Virtual Machines](#)
- [Nucleus Extension Requirements](#)
- [Providing DBA Control of the Nucleus](#)

- [Nucleus Directory](#)

### Allocating Adabas Minidisk Space

An A-Disk of at least 500 4-kilobyte blocks is required, and if no DBA virtual machine exists, either a library minidisk of at least 1500 4-kilobyte blocks is needed, or a read only LINK to another nucleus virtual machine containing the library minidisk. Refer to the section [Adabas Library Requirements](#).

Typically, the database resides on z/VM minidisks defined in the CP directory of the Adabas nucleus virtual machine. These minidisks must have multi-write passwords in the CP directory. See the section [Disk Space Requirements for the Database](#).

### Communicating with Other Virtual Machines

To authorize the nucleus' use of z/VM IUCV, place the following statement in the CP directory:

```
IUCV ALLOW (PRIORITY)
```

—where “PRIORITY” is optional.

If access to the nucleus machine is to be restricted, the IUCV statement should be in the CP directory of the user machines. In addition, the CP directory must have the MAXCONN parameter of the OPTION statement set high enough to accommodate one path to each z/VM or guest operating system's user machine and two paths to the ID table manager virtual machine. For example:

```
OPTION MAXCONN 5
```

—would be sufficient to support three Adabas user machines and the ID table manager machine.

Before the Adabas nucleus is started, the following commands must also be issued:

```
ACCESS cuu fm/A
EXEC SETTXTLB
SET STORECLR ENDCMD
—where
cuu is the virtual unit address of the library minidisk.
fm is the z/VM filemode.
```

If used, the online installation procedure creates these statements in the nucleus user machine's PROFILE EXEC file.

## Nucleus Extension Requirements

The Adabas nucleus machine requires four nucleus extensions with the following attributes:

```
ADARUN USER SERVICE
DATADEF SYSTEM
DISPDD SYSTEM
RELDD SYSTEM
```

The NUCXTNTS EXEC may be used to load the four nucleus extensions. If the online installation procedure is used, it adds a statement in the PROFILE EXEC to invoke the NUCXTNTS EXEC when the nucleus virtual machine starts.

If a new version of one of the nucleus extensions is to be activated the following commands must be entered:

```
ERASE extname MODULE A
NUCXDROP extname
NUCXTNTS
```

—where *extname* is the nucleus extension name. If NUCXTNTS is invoked without the two previous commands, the old version of the program remains active.

## Providing DBA Control of the Nucleus

The DBA virtual machine can be defined to z/VM as a secondary console for the Adabas nucleus machine. This allows the DBA to control the nucleus machine, and issue Adabas operator commands using the SEND command from a terminal. The secondary console support is defined by specifying the CONSOLE statement in the CP directory of the Adabas nucleus machine as follows:

```
CONSOLE 009 devtype T dbavmid
—where
devtype is the DBA console device type.
dbavmid is the DBA virtual machine ID.
```

## Nucleus Directory

The following is an example of the entries in the CP directory for the Adabas nucleus virtual machine running in ESA mode:

```

USER ADA00001 ADA00001 6MB 64M G
ACCOUNT xx xxxxx
IPL CMS PARM AUTOCR
MACHINE ESA
OPTION MAXCONN 100
IUCV ALLOW PRIORITY
CONSOLE 009 3215 T SAGDBA
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 A

```

The LINK and MDISK statements are as follows:

```

LINK MAINT 190 190 RR
LINK MAINT 19E 19E RR
.
.
MDISK 191 3380 067 005 vvvvvv MR rpassword wpassword mpassword
MDISK 200 3380 304 016 vvvvvv MW rpassword wpassword mpassword
MDISK 300 3380 400 051 vvvvvv MW rpassword wpassword mpassword
MDISK 400 3380 451 021 vvvvvv MW rpassword wpassword mpassword
MDISK 410 3380 472 011 vvvvvv MW rpassword wpassword mpassword
MDISK 420 3380 483 006 vvvvvv MW rpassword wpassword mpassword
MDISK 430 3380 489 002 vvvvvv MW rpassword wpassword mpassword
MDISK 431 3380 491 002 vvvvvv MW rpassword wpassword mpassword
MDISK 440 3380 493 002 vvvvvv MW rpassword wpassword mpassword
MDISK 441 3380 495 002 vvvvvv MW rpassword wpassword mpassword

```

The ECMODE OPTION statement, though not required, is recommended because it enables the use of certain VM assists.

## ID Table Manager Virtual Machine

The ID table manager executes in its own virtual machine, which normally runs disconnected. For information about the function of the ID table manager, refer to the section [Functions of the ID Table Manager](#).

The virtual storage size required for the ID table manager is four megabytes. The A-disk must be at least 100 4-kilobyte blocks and have a read-only link to the Adabas library disk (where the ADAV<sub>vrs</sub> TXTLIB and all CONTROL files reside).

By default, the ID table manager allows 20 ID table entries. A zap can be used to extend this limit:

```
NAME ADAITM ADAITM
VER 0022 00FF      default 20
REP 0022 00xx      new default
```

- [Communicating with Other Virtual Machines](#)
- [ID Table Manager Directory](#)
- [Functions of the ID Table Manager](#)

### Communicating with Other Virtual Machines

The ID table manager must be authorized for IUCV communication. The following entries should be placed in the CP directory:

```
IUCV ALLOW (PRIORITY)
IUCV ANY (PRIORITY)
```

—where “PRIORITY” is optional, but recommended. In addition, the OPTION statement’s MAXCONN parameter in the CP directory must be set high enough to accommodate an IUCV path to each user virtual machine, either z/VM or a guest operating system, and two paths to each Adabas nucleus machine.

The PROFILE EXEC of the ID table manager virtual machine should contain the following commands:

```
LINK SAGDBA cuu cuu RR rpassword
ACC cuu fm/A
EXEC SETTXTLB
LOAD ADAITM (RESET ADAITM
START * ...
```

—where *cuu* is the virtual unit address of the library minidisk and *fm* is the z/VM filemode.

Software AG recommends starting the ID table manager machine automatically at system startup time using the AUTOLOG command in the PROFILE EXEC of the AUTOLOG1 virtual machine. If set up to do so, the ID table manager virtual machine’s PROFILE EXEC executes AUTOLOG to start the Adabas virtual machines.

The ID table manager accepts parameters for the following values:

- DBA virtual machine ID, keyword DBAVMID;
- ID table manager target ID, keyword NODEID;

- ID table manager node name, keyword `NODENAME`;
- Entire Net-work IUCV resource ID, keyword `NETRESID`.

These keyword parameters are specified in the z/VM `START` command. For example, to execute the ID table manager with a node ID of 1001 and a node name of `CMSNODE`, enter the following `START` command:

```
START * NODEID 1001 NODENAME CMSNODE
```

Except for the `DBAVMID` parameter, these parameters are only relevant to operation with Entire Net-Work.

### ID Table Manager Directory

The following is an example of the entries in the CP directory for the ID table manager virtual machine running in ESA mode:

```
USER DBIDSERV DBIDSERV 6M 8M G
ACCOUNT xxxxxxxx
OPTION MAXCONN 100
IUCV ALLOW PRIORITY
IUCV ANY PRIORITY
IPL CMS ESA PARM AUTOCR
CONSOLE 009 3215 T SAGDBA
SPOOL OOC 2540 READER *
SPOOL OOD 2540 PUNCH A
SPOOL OOE 1403 A
LINK MAINT 190 190 RR
LINK MAINT 19E 19E RR
MDISK 191 3380 2741 vvvvvvv MR rpassword
```

### Functions of the ID Table Manager

Under z/VM, commonly addressable and modifiable storage areas are not available. This means that there can be no centrally located routing table as in other operating systems.

Instead, this information is maintained and distributed by the ID table manager program, `ADAITM`, which executes in its own virtual machine. It is a required virtual machine and operates continuously in disconnected or background mode.

`ADAITM` also provides all information needed for communication between z/VM and any guest operating systems running under VM.

The Adabas nucleus (`ADAMPM`) and `ADALNK` establish an IUCV communications path to the virtual machine running `ADAITM` during the startup procedure. The VMID of the ID table manager virtual machine is at a fixed location in `ADALDI`, a program used by `ADAMPM` and `ADALNK`

for communications. Both this VMID and the target ID of the ID table manager can be set at installation time and should not be changed. The default VMID is DBIDSERV and the default target ID is 65535.

The DBA virtual machine ID, ID table manager target ID, and node name can all be specified in the ID table manager PROFILE EXEC as parameters:

```
LOAD ADAITM ( RESET ADAITM'
START * DBAVMID vm-id NODEID node-id NODENAME node-name
NETRESID res-id
-where
vm-id is the ID of the DBA virtual machine
node-id is the ID table manager target ID
node-name is the ID table manager node name
res-id is the Entire Net-Work IUCV resource ID
```

If desired, the DBA virtual machine ID and ID table manager target ID (node ID) can also be zapped.

With the exception of the DBAVMID parameter, the parameters above are only relevant to operation with Entire Net-Work.

The following zap modifies the default DBID and the database administrator machine VMID (described later) in the ID table manager:

```
NAME ADAITM ADAITM
VER 0020 FFFF
VER 0028 4040,4040,4040,4040
REP 0020 FFFF
REP 0028 E2C1,C7C4,C2C1,4040
```

This zap illustrates how to modify VMID of the ID table manager in ADALDI:

```
NAME ADALDI ADALDI
VER 0034 C4C2,C9C4,E2C5,D9E5
REP 0034 C4C2,C9C4,E2C5,D9E5
```

If an ADAITM program error occurs and the restart option is active, all information concerning the Adabas z/VM environment is written to the z/VM file ADAITM RESTART on the ID table manager's A-Disk and a message is sent to both the CP operator and (if one has been defined) to the database administrator virtual machine consoles. ADAITM then severs all IUCV paths, places a command in the z/VM program stack to reload and restart itself, and stops.

If the restart option is not active, no ADAITM RESTART file is written; the ID table manager logs off, and terminates all active nuclei abnormally.

During the time that ADAITM is not available, existing local communication paths may still be used; however, no new paths can be initiated and no new Adabas nuclei can be started. All remote communications are terminated.

After restarting, ADAITM reconnects to all Adabas z/VM nuclei and to any Adabas users in z/VM machines who are still active. If another error occurs during error recovery or restart, ADAITM sends a message to both the CP operator and the database administrator virtual machine consoles, and then logs off the ADAITM virtual machine. This causes all local Adabas nuclei toabend. Stopping the ADAITM virtual machine with `FORCE` also abends all local nuclei.

ADAITM accepts operator commands for administration purposes. These operator commands comprise two general categories: listing commands and trace commands. Listing commands (`LISTxxxx`) display console lists of the requested information, usually as `AITMnn` messages. Trace commands provide chronological information such as nucleus/user initialization and termination.

The commands are

Command	Description
DISPON	Displays events on the z/VM console as they occur
DISPOFF	Stops event display on the z/VM console
LISTLINK	Displays all active links to directly addressable network nodes (for Entire Net-Work systems only)
LISTLOG	Displays the contents of the logging area on the console
LISTMSG	Displays the contents of the NETITM MSGS file, which contains messages related to Entire Net-Work communication
LISTNODE	Displays all the active network nodes on the console
LISTTARG	Lists active targets (active nuclei, etc.) on the console
LISTUSER	Lists all user virtual machines on the console
LOGON	Records events in a logging area
LOGOFF	Stops logging of events
RESET	Clears the event logging area



## Database Administrator (DBA) Virtual Machine

To give the database administrator (DBA) maximum control of the Adabas environment, a separate DBA virtual machine should be allocated. This allows the DBA to

- perform multi-user utility operations on multiple databases;
- maintain a single Adabas library disk, ADAV<sub>VRS</sub> TXTLIB; and
- issue operator commands for nuclei running in disconnected machines (when the DBA virtual machine is defined as a secondary console for the nucleus machine).

The DBA virtual machine requires a minimum of 4 megabytes of virtual storage. Depending on the Adabas utilities to be run in the DBA virtual machine and the parameters specified, more storage may be required. The standard virtual storage size is 4 megabytes.

- [Allocating DBA Minidisk Space](#)
- [Communicating with Other Virtual Machines](#)
- [DBA Nucleus Extension Requirements](#)
- [DBA Directory](#)

### Allocating DBA Minidisk Space

The A-Disk must be at least 750 4-KB blocks (or the equivalent). If this virtual machine is being used, the Adabas library disk must be defined in the CP directory for the DBA virtual machine. See the section [Adabas Library Requirements](#).

Multiple write links must be defined from the DBA virtual machine to each database minidisk that the DBA machine supports. If the online installation procedure is used and LINK passwords are specified, LINK statements are automatically created in the DBA virtual machine's PROFILE EXEC.

### Communicating with Other Virtual Machines

The DBA virtual machine must be authorized for IUCV communication. The following entry should be placed in the CP directory:

```
IUCV ALLOW PRIORITY
```

The MAXCONN parameter of the OPTION statement in the CP directory must be set high enough to accommodate an IUCV path to the ID table manager and paths to each Adabas nucleus machine.

To execute Adabas utilities, the DBA virtual machine must have the following statements in its PROFILE EXEC:

```
ACCESS cuu fm/A
EXEC SETTXTLB
SET STORECLR ENDCMD
—where
cuu is the virtual unit address of the library minidisk
fm is the filemode of the library minidisk
```

If used, the online installation procedure creates these statements in the PROFILE EXEC.

### DBA Nucleus Extension Requirements

The database administrator virtual machine requires four nucleus extensions with the following attributes:

```
ADARUN USER SERVICE
DATADEF SYSTEM
DISPDD SYSTEM
RELDD SYSTEM
```

The NUCXTNTS EXEC may be used to load the four nucleus extensions. If the online installation procedure is used, it adds a statement to the PROFILE EXEC to invoke the NUCXTNTS EXEC when the DBA virtual machine starts.

If a new version of one of the nucleus extensions is to be activated, enter the following commands with NUCXTNTS:

```
ERASE extname MODULE A
NUCXDROP extname
NUCXTNTS
```

—where *extname* is the nucleus extension name.

If NUCXTNTS is invoked without ERASE and NUCXDROP, the old version of the program remains active.

## DBA Directory

The DBA virtual machine's CP directory requires an `OPTION` statement. The following is an example of the entries in the CP directory for the DBA virtual machine:

```
USER SAGDBA SAGDBA 6M 8M G
ACCOUNT xxxxxxxx
OPTION MAXCONN 10
IUCV ANY
IPL CMS PARM AUTOOCR
```

## User Virtual Machine

Each user virtual machine requires 1536 KB of virtual storage if Natural is to be used without a DCSS. If Natural is installed as a DCSS, 768 KB of virtual storage is sufficient.

- [Allocating User Minidisk Space](#)
- [Communicating with Other Virtual Machines](#)
- [User Nucleus Extension Requirements](#)
- [User Directory](#)
- [ADARUN Control of User Programs](#)

### Allocating User Minidisk Space

Each user virtual machine must `LINK` and `ACCESS` to either the Adabas library disk (that is, where `ADAVvrs` `TXTLIB` resides), or to a sublibrary disk. Application programs executing in `SINGLE` user mode must use the Adabas library disk `ADAVvrs`. A sublibrary disk is created when users are to be isolated from the Adabas library disk. For more user information on the sublibrary disk refer to the section [Creating a User Sublibrary](#).

### Communicating with Other Virtual Machines

The user machine only needs to be authorized to use `IUCV` when the ID table manager and nucleus virtual machines do not have `IUCV ALLOW` statements in their respective CP directory entries. If neither the ID table manager nor the Adabas nucleus virtual machine has the `IUCV ALLOW` statements, issue the following statements in the user CP directory:

```
IUCV idtmvmid
IUCV nucvmid
-where
idtmvmid is the ID table manager ID
nucvmid is the Adabas nucleus virtual machine ID
```

Refer to the section *IUCV Security Options*. The `MAXCONN` parameter of the `OPTION` statement in the CP directory must be set high enough to accommodate an IUCV path to the ID table manager and one path to each Adabas nucleus machine.

To execute ADALNK, the user virtual machine must have the following statements in its PROFILE EXEC:

```
ACCESS cuu fm/A
GLOBAL TXTLIB libname
—where
cuu is the virtual unit address of the library or sublibrary minidisk
fm is the filemode of the library or sublibrary minidisk
libname is the TXTLIB filename
```

### User Nucleus Extension Requirements

A user virtual machine requires four nucleus extensions with the following attributes:

```
ADARUN SYSTEM SERVICE
DATADEF SYSTEM
DISPDD SYSTEM
RELDD SYSTEM
```

The `NUCXTNTS EXEC` must be issued with the parameter `USER` to load the four nucleus extensions.

To activate a new version of a nucleus extension, enter the following commands:

```
ERASE extname MODULE A
NUCXDROP extname
NUCXTNTS USER
```

— where *extname* is the nucleus extension name.

If `NUCXTNTS` is invoked without `ERASE` and `NUCXDROP`, the old version of the `NUCXTNTS` program remains active.

## User Directory

The DBA virtual machine's CP directory requires an OPTION statement as shown in the following example:

```

USER ADAUSER1 ADAUSER1 6M 8M G
ACCOUNT xxxxxxx
IPL CMS PARM AUTOCR
MACHINE ESA
OPTION MAXCONN 10
CONSOLE 009 3215
SPOOL 00C 2540 READER *
SPOOL 00D 2540 PUNCH A
SPOOL 00E 1403 A
LINK MAINT 190 190 RR
LINK MAINT 19E 19E RR
MDISK 191 3380 381 007 vvvvvv MR rpassword

```

## ADARUN Control of User Programs

ADARUN control statements can be used to control user programs by issuing the appropriate DATADEF statement for DDCARD and optionally for DDPRINT. The DATADEF statement assures that all ADALNK errors are both displayed on the virtual console and recorded in the DDPRINT file.

## Releasing the User Virtual Machine's Communication Environment

The Adabas communications environment can be reset and all storage allocated to that environment in a user virtual machine can be released by issuing the following z/VM command:

```
NUCXDROP ADARUN
```

Because the initialization and termination of the communication environment uses a lot of system resource, the NUCXDROP command is recommended only when no further use of Adabas communication from the user virtual machine is planned for the session.

## Adabas Library Requirements

The Adabas library disk must be at least 2700 4-KB blocks (or the equivalent). Each added database after the first adds a requirement of 100 4-KB blocks to the library disk. The library minidisk must be defined with a multiread or ALL password in either the DBA virtual machine's CP directory, or in the nucleus virtual machine's CP directory if no DBA machine has been defined. All virtual machines accessing the database must have read-only LINKs to the library minidisk.

Installations wishing to isolate multiuser application programs from the Adabas library disk can define a sublibrary on a commonly accessible minidisk, as described in the following section [Creating a User Sublibrary](#). Application programs running in SINGLE user mode must use the Adabas library disk.

- [Creating a User Sublibrary](#)

### Creating a User Sublibrary

A sublibrary disk is created to isolate users from the Adabas library disk. The sublibrary disk must be at least 200 4-KB blocks, be located on a commonly accessible minidisk, and have a multiple-read password. The sublibrary TXTLIB must contain the following members:

```
$SAGIOS ADALNK NETPARS NIUDEP
ADAIOR ADARUN NETRQM NIUEXT
ADAIOS ADAUSER (ADABAS) NETTQM
ADAILD NETBPM NETTRC
ADALDI NETIUCV NETTRT
```

To copy these members from the Adabas library disk, issue the following commands for each member:

```
FILEDEF IN DISK ADAVvvv TXTLIB fm (MEMBER memname
FILEDEF OUT DISK memname TEXT fmwork
MOVEFILE IN OUT
—where
fm is the filemode of the Adabas library disk
fmwork is the filemode of a work minidisk
memname is the name of the member to be copied to the sublibrary.
```



**Note:** ADAUSER has the member name ADABAS.

To create the new TXTLIB sublibrary, enter the following command:

```
TXTLIB GEN sublib memname1,memname2,....memnamex
-where
sublib is the name of the sublibrary
memname1 – memnamex are the names of members taken from the ADAVvrs TXTLIB.
```

After TXTLIB has been created, the text files can be deleted from the work minidisk. It is important to remember that a sublibrary needs separate maintenance; this means that zaps and new SM levels must be applied to both the Adabas library and each sublibrary. The NUCXTNTS, DEFNUCX, and SETXTLB EXEC routines must also be copied to the new sublibrary.

The following additional modules must be generated and then copied to the TXTLIB sublibrary minidisk:

```
ADAIOR DATADEF
ADALNK DISPDD
ADARUN RELDD
```

The z/VM module files having the same names should also be copied. You must also ensure that the sublibrary minidisk is updated whenever the main library minidisk is changed.

Each user machine utilizing the sublibrary minidisk must have the appropriate LINK, ACCESS and GLOBAL TXTLIB statements active. See the section [User Virtual Machine](#).

### Disk Space Requirements for the Database

The database can be on either DASD having MVS or VSE VTOCs, or on VM minidisks. For MVS or VSE disks, there must be multi-write LINKs or ATTACH commands to those disks in the PROFILE or CP directory of the nucleus virtual machine. For VM minidisks, the nucleus virtual machine's CP directory must contain the MDISK commands, and multi-write passwords must be defined.

The Adabas database requires at least five VM minidisks, as follows:

File	Suggested Unit Address	Required Space in Blocks	3380 Cylinders	3350 Cylinders
ASSO	200	8640 (2k)	30 + 2	35 + 1
DATA	300	10800 (4k)	70 + 2	85 + 1
WORK	400	1650 (4k)	10 + 2	15 + 1
TEMP	410	2250 (4k)	15 + 1	20 + 1
SORT	420	2250 (4k)	15 + 1	20 + 1

Optional minidisks and their recommended sizes and requirements are:

File	Suggested Unit Address	Required Space in Blocks	3380 Cylinders	3350 Cylinders
CLOG1	430	300 (4k)	1 + 1	2 + 1
CLOG2	431	300 (4k)	1 + 1	2 + 1
PLOG1	440	300 (4k)	1 + 1	2 + 1
PLOG2	441	300 (4k)	1 + 1	2 + 1

All sizes specified for the minidisks must be rounded up to the next full multiple of cylinders. For FBA devices, see the section [FBA Devices](#).

The first cylinder or FBA pseudo-cylinder of each database minidisk is reserved for the z/VM directory. Therefore, one cylinder must be added to each minidisk allocated in the CP directory. The last column of each of the tables above accounts for the directory cylinder by adding one to the normal cylinder/pseudo-cylinder count.

The z/VM `RESERVE` command must be specified for each of the database minidisks. If the online installation procedure is being used, this `RESERVE` operation is performed when the installation procedure is executed.

Any Adabas virtual machine that executes multiuser utilities must define multi-write LINKs to the database minidisks.

---

## Adabas File Support under z/VM

- [DASD Supported by Adabas](#)
- [Sequential File Support](#)
- [Using Improved Data Recording Facility \(IDRC\) Tapes](#)
- [Tape Management with User Exit CMSUX1](#)
- [DATADEF File Assignments](#)
- [DATADEF Error Codes](#)

### DASD Supported by Adabas

Adabas z/VM supports direct access database files on z/VM-formatted disks or on CKD and FBA disks with an MVS/VSE VTOC. The disks can be minidisks or real disks. The files may be contained on one or more real or virtual volumes.

z/VM-formatted disks must not be attached with the CP `ATTACH` command, but must be defined in the directory of the virtual machine or linked to it using the CP `LINK` command. Disks with database files need not be accessed using the z/VM `ACCESS` command. If a file spans multiple volumes, all of the devices must be of the same type and must have the same format; that is, either all devices must have an MVS or VSE VTOC, or all must be z/VM-formatted.



## Sequential File Support

Sequential files are supported as

- terminal files (input/output)
- card reader files (input only)
- card punch files (output only)
- printer files (output only)
- tape files (input/output)
- z/VM-formatted disk files (input/output)
- MVS- or VSE-formatted disk files (input only)

The file format for terminal files must be fixed length with a maximum record length of 80. When an input file is assigned to terminal, the user is prompted with the name of the file being read. The end of a terminal input file is signaled by pressing `ENTER` without any input.

Sequential files on other unit record devices must also be fixed length. The maximum record length is the length supported by z/VM on the device.

All record formats and lengths are supported for tape files. If the tape file was created under an operating system which does not create HDR 2 records, the user must supply file format and length information. Tape files must reside on standard labeled volumes. The `z/VM TAPE` command may be used to initialize a tape volume and set its density. To avoid confusion during tape handling, it is highly recommended that each tape be labeled with a unique 6-byte name. Both multi-file volumes and multi-volume files are fully supported.

Sequential files on z/VM-formatted disks may be either fixed or variable length. Blocking is done by the z/VM file service routines. Unlike DASD volumes used for direct access files, disks containing sequential files must be `ACCESSed`. The z/VM file services do not differentiate between nonexistent files and empty files. To prevent open errors in Adabas utilities when a non-existent input file is used, a one-byte record containing a dollar sign (\$) is written to sequential files on z/VM formatted disks at close time if no records have been written.

All record formats and lengths supported by MVS or VSE may be used on sequential MVS/VSE DASD files. If there is no record information in the label records (VSE), record format and length must be specified by the user. MVS/VSE DASDs may not be used for sequential output files.

Adabas supports concatenation of sequential files if the record format and the logical record lengths of the concatenated files are the same. Concatenation of files on different storage media is also possible.

## Using Improved Data Recording Facility (IDRC) Tapes

Sequential files to be read backward can be on either tape or z/VM-formatted disks. Exceptions to this are tape files created using the improved data recording capability (IDRC) on certain cassette units. When this facility is in use on certain other models of tape units, the read backward channel command is not supported. This point should be considered when planning recovery strategies for rebuilding the database after a possible failure.

Where supported, sequential files to be read backward may also be concatenated. If they are concatenated, they must be specified in the same order as though they were being processed normally.

The tape units to be used must be attached before file assignment by DATADEF unless they are to be dynamically allocated at open time by a tape management system. In this last case, a generic name can be specified in the DATADEF statement (see the section *DATADEF File Assignments*).

Tape units can be attached to any virtual address, and need not have any logical name. During the open process, Adabas reads the first block of the tape to verify that the correct volume is mounted. If the correct volume is not mounted or is not correctly initialized, the message ADAI48 is issued and the tape is unloaded. Message ADAI40 is then issued to the operator, and message ADAI41 is sent to the CP operator.

When a tape volume is successfully mounted, the message ADAI42 is displayed on the z/VM virtual console. This message also indicates whether the IDRC is in effect. If a cassette created with IDRC is used on a unit which does not support IDRC, an I/O error occurs.

If the volume parameters were supplied to DATADEF, the required volume serial number is included in both messages; if an output file is to be created and no volume serial number has been specified, the constant SCRTCH is displayed. This means that any unused standard label work (commonly called "scratch") tape should be mounted. The two messages are repeated once each minute until the mount request has been satisfied. To reset the mount request, the z/VM operator must press the ENTER key twice. This causes open processing for the file to be terminated abnormally, and is equivalent to having defined the file as 'dummy'.

## Tape Management with User Exit CMSUX1

You can control tape file management with a program that uses the optional user exit CMSUX1. If a program with the name CMSUX1 is loaded with the z/VM LOAD command before invoking a utility or nucleus that requires a tape file, the CMSUX1 program is called

- when the tape file is opened;
- whenever a tape volume must be mounted; and
- when the file is closed.

If a tape is rejected after a mount request because of a missing VOL1 record or incorrect volume/serial number, the CMSUX1 program is also reinvoked before a mount request is repeated.

When the CMSUX1 user exit receives control, general register 0 contains one of the following function codes:

```
1 for open
2 for tape mount
3 for close
```

General register 1 contains the address of a parameter list, which is passed to the CMSUX1 user exit program. The parameters are

- address of the data definition block (DDB) (non-modifiable);
- address of the six-byte tape volume serial number which will be requested (modifiable);
- address of a fullword containing:
  - an input/output flag byte (X'00' for input files, or X'80' for output files);
  - a reserved byte;
  - a two-byte hexadecimal tape unit address in the format X'0 $ccu$ ' (ESA: X' $ccu$ '). If the DATADEF statement specified TAP $x$  in the UNIT parameter, this field contains the corresponding "18 $x$ " value according to standard z/VM tape unit conventions.
- address of a fullword containing the number of times the mount has been attempted (non-modifiable).

If the volume serial number has been modified, the new volume serial number is requested. If the file is an input file, however, this modification is not reflected in the DDB.

Upon return to the system, one of the following return codes is expected in general register 15:

```
0 proceed with the mount processing
4 proceed with the mount processing without issuing any operator messages
8 abort the mount processing
```

If return code 8 is encountered, the mount operation terminates as if an I/O error had occurred.

A sequential system file with the name of DUMP is available to facilitate dumping. This file may be assigned to printer, a z/VM file or DUMMY. If the file is DUMMY a CP dump will not be taken when an error occurs.

## DATADEF File Assignments

All assignments for files which are accessed by an Adabas nucleus or utility must be done using DATADEF, which replaces the z/VM FILEDEF command in the Adabas environment. DATADEF accepts parameters as either a tokenized or extended parameter list; the extended parameter list takes precedence.

The file assignments established by the DATADEF statements can be listed using the DISPDD program. The file name of a specific DATADEF statement can be entered as a DISPDD parameter; if done, only the information for that file is displayed. If no parameter is specified, information for all assigned files is displayed.

The program RELDD can be used to clear active DATADEF entries. RELDD accepts a list of file names to be released or—if no list is specified—clears all active DATADEF entries.

The DATADEF statement creates a data definition block (DDB), which remains in system storage until it is

- overwritten by another DATADEF statement with the same name;
- cleared by RELDD; or
- cleared by a z/VM IPL.

The parameters for DATADEF consist of one positional parameter and one or more keyword parameters separated by commas. An equal sign (=) must be used between a keyword and the parameter value. Depending on how DATADEF is invoked, spaces may be required surrounding equal signs, commas, and parentheses.

The DATADEF parameters are described in the following table:

Parameter Keyword	Required/ Optional	Maximum Length	Specifies . . .
positional	Required	8	the file name (DD) names as specified in <i>Adabas Operations</i> or <i>Adabas Utilities</i> .
BLKSIZE	See note 2	5	the length of the physical blocks in the file. If RECFM=FB, BLKSIZE must be an integral multiple of LRECL; if RECFM=V or RECFM=VB, BLKSIZE must be at least equal to LRECL + 4.
BUFNO	Optional	3 (1 - 255)	the number of buffers to be allocated for a sequential file on tape. Default: 5
COMPRESS	Optional	3 (YES - NO)	whether or not an output file on tape should make use of IDRC available on certain cassette units. If IDRC is not supported, this parameter is set to NO, the default.
CONCAT	Optional	3 (1 - 255)	a concatenation sequence number for the file. This results in the DDB being concatenated to another existing DDB with the same file name. The sequence numbers must be specified in ascending order with no numbers left out. The first file

Parameter Keyword	Required/ Optional	Maximum Length	Specifies . . .
			to be concatenated has the number 1. If specified for a file to be read backward, the sequence numbers are to be given in the normal sequential order. A DATADEF statement without CONCAT frees any existing root DDB and any DDBs concatenated to it.
DISP	Optional for output only	3	whether a sequential file is to be created (NEW), extended (MOD) or overwritten (OLD). If NEW is specified and the file exists a return code is issued. If OLD is specified, the file's existence is not checked. Default: OLD
DSN	Required if not dummy	44	the dataset or SFS directory name.
DUMMY	Optional	-	that the file does not exist. DUMMY is a keyword without a value. It may not be specified with any other parameter except file name.
EXTEND	Optional	-	that an existing DDB is to be extended. EXTEND is a keyword without a value. It may only be specified with the file name, VOL, and UNIT parameters.
FILESEQ	Tape only, optional	3	the sequence number of the file on a multi-file tape. The default value (1) is required if tapes are to be read backward.
FNAME	Required if DSN is a SFS	8	the file name of an SFS member.
FTYPE	Required if DSN is a SFS	8	the file type of an SFS member.
LRECL	See note 2	5	the length of the physical blocks in the file. If RECFM=FB, BLKSIZE must be an integral multiple of LRECL; if RECFM=V or RECFM=VB, BLKSIZE must be equal to or more than LRECL + 4.
MODE	See note 1	2	the z/VM filemode.
RECFM	See note 2	2	the format of the records in the file (F, FB, V, VB, U).
UNLOAD	Tape only, optional	3 (YES - NO)	whether or not the tape is rewound and unloaded. If NO is specified, the tape is rewound at close but is not unloaded. Default: YES.
UNIT	See notes 1 and 3	See note 1	a list of virtual addresses ( <i>cuu</i> , or <i>ccuu</i> for ESA) of the unit or units containing the file, or one of the logical device abbreviations: TRM, PUN, RDR, PRT, or SFS. If a unit address list is given, it must be enclosed in parentheses and entries must be separated by commas.
VOL	See note 1	See note 1	a list of the serial numbers (each at most 6 characters) of the volumes containing the file; if multiple volumes are specified, they must be separated by commas and enclosed in parentheses.

*Notes:*

1. A `MODE` parameter is required for sequential z/VM DASD files. For DASD volumes containing database files, either a `VOL`, `UNIT`, or `MODE` parameter is required. If the database file spans multiple volumes, `VOL` or `UNIT` must be specified. Specifying `MODE=*` for a non-existent file results in a return code of 32.

If both `VOL` and `UNIT` are specified, the number of volumes and unit addresses must be equal and each volume in the `VOL` list must be mounted on the unit specified by the corresponding entry in the `UNIT` list (the first `VOL` entry must be mounted on the first `UNIT` entry, and so on).

For tape files, a `UNIT` must be specified and a real unit attached prior to `DATADDEF` execution, unless the tape unit is dynamically allocated at open time; in this case, `TAPx` can be specified according to standard z/VM conventions, where `TAP1` specifies virtual unit 181, `TAP2` specifies unit 182, and so on. Only one tape unit address is allowed in the `UNIT` parameter.

The `VOL` parameter is required for input tape files, but is optional for output tape files. If a tape file spanning multiple volumes is to be read backwards, specify the volumes in the normal sequential order.

When creating a multi-volume tape file, Adabas z/VM maintains a list of the file volumes. To refer to that volume list in a later `DATADDEF`, specify `VOL=*filename` where `filename` is the name of the multi-volume file.

2. The parameters `RECFM`, `LRECL`, and `BLKSIZE` are required only for tape input files without a `HDR2` label and for `VSE` sequential DASD files. If `RECFM` has been specified, the corresponding `BLKSIZE` and `LRECL` parameters are also required.
3. If `UNIT=SFS`, the `DSN` parameter is used as the `SFS` directory name and the `FNAME` and `FYPTPE` parameters are used as the file name and file type, respectively. For example:

```
DATADDEF DDCARD,DSN=SFSPool:USERID.DTR1,UNIT=SFS,FNAME=DB52,FYPTPE=RUN1
or
DATADDEF DDCARD,DSN=.DTR1,UNIT=SFS,FNAME=DB52,FYPTPE=RUN1
or
DATADDEF DDSAVE1,DSN=.DTR1,UNIT=SFS,FNAME=SSF,FYPTPE=RUN001
```

## DATADEF Error Codes

The following error codes may be returned by DATADEF:


Response Code	Description
16	No parameter list was supplied
20	Invalid keyword
24	No file name specified
28	Error in DSN or DUMMY specification: neither a DSN nor DUMMY was specified; or conflicting parameter specification for a dummy file
32	Error in VOL, UNIT or MODE parameter
36	Incorrect length for VOL , UNIT or MODE parameter
40	Insufficient virtual storage
44	Internal error issuing a CP command
48	Invalid <i>cuu</i> address (internal error)
52	Volume or unit not available or non-VTOC volume has been attached
56	Database file resides on volumes with mixed formats (VTOC, non-VTOC)
60	Database file resides on volumes of different device types
64	More than one unit specified for a tape file
68	Invalid file sequence number
72	Invalid RECFM parameter
76	Invalid BLKSIZE parameter
80	Invalid LRECL parameter
84	Invalid DISP parameter
88	Invalid concatenation count
92	Invalid DDB extension

These codes can be returned by invoking the CODES EXEC with the error number as a parameter.

## Entering Operator Commands

Adabas operator commands can be entered directly from the virtual machine console of either the Adabas nucleus virtual machine, or from another virtual machine console authorized as a secondary console.

In addition, certain z/VM commands can also be entered as Adabas operator commands. This feature allows the database administrator to perform certain display or query functions while the Adabas nucleus is active. To enter a z/VM command, enter the command with the prefix CMS, followed by at least one blank before the command.

-  **Caution:** If issued while Adabas is active, certain z/VM commands can have adverse effects on the nucleus, causing abnormal operation and even the erroneous data. Note also that issuing a z/VM command causes the nucleus to stop operation temporarily while the command is being processed, making Adabas unavailable during that time. The use of this z/VM command facility is solely the responsibility of the user; Software AG cannot accept responsibility for damage or loss that may occur when using this facility

## Setting Defaults for ADARUN

---

Default values for the ADARUN parameters device type and database ID can be zapped into ADARUN at the offsets X'5D8' and X'8E8', respectively. For convenience, a file containing sample zap control statements to modify ADARUN has been supplied on the release tape. The file is named ADARUN ZAP. After making the necessary changes to this file, the user can enter the following commands to modify ADARUN:

```
ACC cuu fm
ZAP TXTLIB ADAVvvv (INPUT ADARUN
ACC cuu fm/A
—where
cuu is the virtual unit address of the Adabas library minidisk
fm is the z/VM filemode
```

## Installing New SM Levels

---

New SM levels may be installed by performing the following steps:

▶ **to install new SM levels:**

- 1 Attach a tape unit to the database administrator virtual machine and mount a scratch tape.
- 2 Back up the existing library disk:

```
TAPE DUMP * * fm (fm is the filemode of the Adabas library disk)
TAPE RUN
```

- 3 Mount the system maintenance tape.



- 4 Enter the following commands:

```
ACC cuu fm (cuu is virtual unit addr, fm is the filemode of the Adabas library
disk)
SMADA
```

- 5 If you have either created a user sublibrary, or if you have the ADARUN, DATADEF, RELDD or DISPDD modules on a commonly accessible minidisk, you must refresh the respective sublibrary and/or minidisk.

## Applying Zaps in z/VM

In the z/VM environment, a zap should be applied using the ZAP command, to verify and replace data. The zap input control statements are in MVS ZAP format. The ZAP command will apply the zap to a member of a z/VM TXTLIB.

The format of the ZAP command is:

```
ZAP TXTLIB libname (INPUT filename PRINT
```

—where

*libname* is the Adabas TXTLIB (ADAV*vrs*)

*filename* is a z/VM file with a filetype of ZAP that contains the input control statements.

The following are examples of ZAP control statements.

```
NAME membername csectname
VER disp data
REP disp data
END
LOG fixnum ZAPLOG text
* (COMMENT)
```



**Note:** In the verify (VER...) and replace (REP...) statements, commas are acceptable data separators. However, commas with spaces or spaces alone are not acceptable data separators and can result in errors.

Refer to the *IBM z/VM Operator's Guide* for information about the ZAP service program.

## IUCV Security Options

---

The *inter-user communications vehicle* (IUCV) is a communication facility that allows one virtual machine to communicate with another virtual machine. Installations control the use of IUCV through the virtual machine directory entries.

In the Adabas environment, security on IUCV communications can be implemented by supplying different IUCV control statements. The following are examples of IUCV security options where:

IUCV ALLOW	is a general authorization indicating that any other virtual machine may establish a communications path with this virtual machine. No further authorization is required in the virtual machine initiating the communication.
IUCV ANY	is a general authorization indicating that a communications path can be established between this virtual machine and any other virtual machine.
IUCV <i>userid</i>	is the one- to eight-character user identification of the virtual machine with which this virtual machine is authorized to communicate.
<i>idtmvmid</i>	is the VMID of the ID table manager.
<i>nucvmid</i>	is the VMID of the Adabas nucleus.

### No Security

```
ID Table Manager: IUCV ALLOW (PRIORITY)
IUCV ANY (PRIORITY)
Adabas Nucleus: IUCV ALLOW (PRIORITY)
DB Administrator: IUCV ALLOW (PRIORITY)
User: No IUCV CP directory entry required
```

### Security on One Database

```
ID Table Manager: IUCV ALLOW
IUCV ANY
Adabas Nucleus: No IUCV CP directory entry required
DB Administrator: IUCV ALLOW
IUCV nucvmid
User: IUCV nucvmid
```

## Total Security

```
ID Table Manager: IUCV ANY
IUCV ALLOW
Adabas Nucleus: IUCV idtmvmid
DB Administrator: IUCV idtmvmid
IUCV nucvmid
User: IUCV idtmvmid
IUCV nucvmid
```

## Considerations for Installing ADALNK/ADAUSER

- [ADALNK Considerations](#)
- [User Exit B \(Pre-Command\) and User Exit A \(Post-Command\)](#)
- [LNKUES for Data Conversion](#)
- [Creating the Adalink Module \(ADALNK\)](#)
- [ADAUSER Considerations](#)

### ADALNK Considerations

Since link routines are dynamically loaded in most environments, it should only be necessary to replace the existing Adabas TXTLIB or TEXT files with the new Adabas TXTLIB, ADAV*vrs*. Programs that are either GENMODed or in a discontinuous shared segment (DCSS) must be regenerated.

### User Exit B (Pre-Command) and User Exit A (Post-Command)

One or two user exits may be linked with ADALNK:

- UEXITB receives control before a command is passed to a target.



**Note:** Special commands emanating from utilities and from Adabas Online System are marked as physical calls. These calls must be bypassed in user exits. These calls have X'04' in the first byte (TYPE field) of the command's Adabas control block (ACB). UEXITB must check this byte and return if it is set to X'04'. Be sure to reset R15 to zero on return.

- UEXITA receives control after a command has been completely processed.

The user exits must be specified in the LOAD command to be active.

At entry to the exit(s), the registers contain the following:

Register	Contents
1	Address of the UB.  If the flag bit UBFINUB is reset, the contents of the halfword at Adabas + X'86' have been moved to UBLUINFO. If those contents are greater than zero, the two bytes starting at UBINFO (UB+X'40') have been set to zero.  If UBFINUB is set, no changes can be made to the UB or ACB (except for ACBRSP).
13	Address of an 18-word save area (for non-CICS Adalink exits)
14	Return address
15	Entry point address: UEXITB or UEXITA

Any registers except register 15 that are modified by the user exits must be saved and restored; the address of a save area for this purpose is in register 13.

If at return from UEXITB register 15 contains a value other than zero (0), the command is not sent to the target but is returned to the caller. The user exit should have set ACBRSP to a non-zero value to indicate to the calling program that it has suppressed the command: response code 216 is reserved for this purpose.

The UEXITB exit may set the UB field UBLUINFO to any lesser value, including zero; an abend occurs if the user exit sets UBLUINFO to a greater value. The UBLUINFO length cannot be changed when any other exit is used; for example, Adabas Review.

The user information received by a UEXITA exit may have been modified; this modification may include decreasing its length, possibly to zero, by any of the ADANUC user exits.

ADALNK can return the following non-zero response codes in ACBRSP:

Response Code	Description
216	UEXITB suppressed the command
218	No UB available

The following two EQUates, described at the beginning of the source, can be modified before ADALNK is assembled. Other Adalinks allow this information to be zapped.

Equate	Description
LOGID	The default logical ID, ranging in value from 1 to 65535. The default is 1.
LNUINFO	The length of the user information to be passed to Adalink user exits, ranging in value from 0 to 32767. The default is 0.

## LNKUES for Data Conversion

The module LNKUES provides Universal Encoding Support (UES). This module must be linked into the standard batch ADALNK. LNKUES converts data in the Adabas buffers and byte-swaps, if necessary, depending on the data architecture of the caller.

Prior to Version 7, Entire Net-Work converted all data for mainframe Adabas. When Entire Net-Work Version 5.5 and above detects that it is connected to a target database that converts data, it passes the data through without converting it.

LNKUES is called only on ADALNK request (X'1C') and reply (X'20') calls if the first byte of the communication ID contains X'01' and the second byte does not have the EBCDIC (X'04') bit set.

- For requests, LNKUES receives control before UEXITB.
- For replies, LNKUES receives control after UEXITA.

By default, two translation tables are linked into LNKUES/ADALNK:

- ASC2EBC: ASCII to EBCDIC translation; and
- EBC2ASC: EBCDIC to ASCII translation.



**Note:** It should only be necessary to modify these translation tables in the rare case that some country-specific character other than "A-Z a-z 0-9" must be used in the Additions 1 (user ID) or Additions 3 field of the control block.

If you prefer to use the same translation tables that are used in Entire Net-Work:

- in ASC2EBC and EBC2ASC, change the COPY statements from UES2ASC and UES2EBC to NW2ASC and NW2EBC, respectively.
- re-assemble the translation tables and relink LNKUES/ADALNK.

Both the Adabas and Entire Net-Work translation table pairs are provided in the section [Translation Tables](#). You may want to modify the translation tables or create your own translation table pair. Be sure to (re)assemble the translation tables and (re)link LNKUES/ADALNK.

```
//LINK EXEC PGM=IEWL,REGION=0M
// PARM='XREF,REUS,LIST,LET,NCAL,SIZE=(1024K,256K)'
//SYSPRINT DD SYSOUT=*
//SYSLMOD DD DISP=SHR,DSN=USER.LOAD(ADALNK)
//ADALIB DD DISP=SHR,DSN=ADABAS.LOAD
//SYSLIN DD *
MODE AMODE(31) RMODE(ANY)
ENTRY ADABAS
INCLUDE ADALIB(ADALNK)
INCLUDE ADALIB(LNKUES)
INCLUDE ADALIB(ASC2EBC)
INCLUDE ADALIB(EBC2ASC)
NAME ADALNK(R)
/*
//
```

The (re)linked ADALNK must be made available to Entire Net-Work. If you are calling Adabas version 7 and you do not have the correct LNKUES/ADALNK module, Adabas produces unexpected results: response code 022, 253, etc.

### Creating the Adalink Module (ADALNK)

Perform the following steps to create the ADALNK module:

► **to create the ADALNK module:**

- 1 assemble ADALNK and UEXITA/UEXITB

To assemble ADALNK, enter the following z/VM commands:

```
ACC cuu1 filemode/A
ACC cuu2 S
GLOBAL MACLIB ADAV7vv MVSXA HCPGPI DMSGPI DMSOM
ASMAHL ADALNK
-where
cuu1 is the virtual unit address of the Adabas library minidisk
cuu2 is the virtual unit address of the system minidisk containing the MVSXA
MACLIB
filemode is the filemode for the Adabas library minidisk
```

To assemble UEXITA/UEXITB, enter the same commands as above:

```
ACC cuu1 filemode/A
ACC cuu2 S
GLOBAL MACLIB ADAV7vv MVSXA HCPGPI DMSGPI DMSOM
ASMAHL uexitname
—where
cuu1 is the virtual unit address of the Adabas library minidisk
cuu2 is the virtual unit address of the system minidisk containing the MVSXA
MACLIB
filemode is the filemode for the Adabas library minidisk
uexitname is the UEXITA/B filename (with a filetype of TEXT)
```

- 2 load ADALNK and the user exits, by entering the following z/VM commands:

```
LOAD ADALNK (RESET ADALNK RLD
INCLUDE uexitname (RESET ADALNK RLD
```

—where *uexitname* is the filename (with filetype of TEXT) of UEXITA or UEXITB. To load both UEXITA and UEXITB, specify the `INCLUDE` command again for the second user exit.

- 3 generate a module, by entering the z/VM command:

```
GENMOD
```

## ADAUUSER Considerations

ADAUUSER is a program that links the user to Adabas. It is specific to an operating system and is independent of release level and mode.

ADAUUSER operates in the following way:

- ADAUSER is the only program that contains the entry point ADABAS. ADAUSER is dynamically loaded from the Adabas library, ADAV<sub>vvv</sub> TXTLIB, when a user program is loaded.
- On the first call to Adabas, ADAUSER invokes ADARUN, which is installed as a SYSTEM nucleus extension at LOGON time by the NUCXTNTS EXEC, and is release-independent. Subsequent Adabas calls bypass ADARUN.
- When invoked, ADARUN processes its control statements. If the ADARUN PROGRAM parameter has the (default) value USER, ADARUN loads ADALNK if the ADARUN MODE parameter specifies MULTI (the default), or ADANUC if the ADARUN MODE parameter specifies SINGLE. Therefore, ADAUSER is mode-independent.





# 5 Device and File Considerations

---

- FBA Devices ..... 46
- Adding New Devices ..... 47
- General Rules for Defining Device Block Sizes ..... 50

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

- installing on fixed-block addressing (FBA) devices; and
- defining new devices.

## FBA Devices

---

All device definitions for Adabas control statements for FBA disks should specify one of the following devices types:

- FBA SCSI devices: Specify device types of 5512, 6512, or 7512.
- Virtual FBA devices: Specify device types of 5121, 5122, or 5123.



**Note:** Virtual FBA devices are not permanent and are, therefore, only suitable for holding temporary or work data sets.

Choose a device type based on the block sizes given in the following tables:

### SCSI Device Types:

Dev Type	Asso blkz	Data blkz	Work blkz	Temp blkz	Sort blkz	PLOG blkz	CLOG blkz
5512	2048	4096	4096	4096	4096	4096	4096
6512	4096	8192	8192	8192	8192	8192	8192
7512	4096	16384	16384	16384	16384	16384	16384

### Virtual FBA Device Types:

Dev Type	Asso blkz	Data blkz	Work blkz	Temp blkz	Sort blkz	PLOG blkz	CLOG blkz
5121	2048	4096	4096	4096	4096	4096	4096
5122	4096	8192	8192	8192	8192	8192	8192
5123	4096	16384	16384	16384	16384	16384	16384

The pseudo-cylinder for each of these devices has a different number of blocks as described below:

```
5512 cylinder = FBA blocks/1024
6512 cylinder = FBA blocks/1024
7512 cylinder = FBA blocks/1024
5121 cylinder = FBA blocks/960
5122 cylinder = FBA blocks/960
5123 cylinder = FBA blocks/960
```

The size definitions for FBA devices on Adabas control statements can specify the number of pseudo-cylinders or the number of Adabas blocks (RABNs).

Make sure that the number of FBA blocks in the minidisk is a multiple of the pseudo-cylinders listed above. In addition, the minidisk size must be one pseudo-cylinder larger than the size specified in the Adabas size definitions:

- An SCSI pseudo-cylinder comprises 1,024 elements of 512 bytes each, or 512 K per pseudo-cylinder.
- A virtual FBA pseudo-cylinder comprises 960 elements of 512 bytes each, or 480 K per pseudo-cylinder.

## Adding New Devices

Support for new device types that include user-defined block sizes can be implemented in Adabas 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).

Under z/VM 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 and 7.1, 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'680'.

This information is valuable when adding an additional TDCE entry.

The ADDEVICE EXEC distributed on the release tape can be used to add new device types. This EXEC creates zap files called USERDEV $n$  ZAP to modify the TDCEs. It also modifies the table of valid device names DEVICE TABLE and the USERDEV TABLE. These tables are used for verification during the online installation procedure.


- Information to be Zapped into the First Free TDCE

### Information to be Zapped into the First Free 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'28' (reserved for BS2000 device type 2006), X'2B', or X'2E'
TDCF	03	The flag bit must be set—TDCFFBA (X'80') for FBA/PAM devices or TDCFCKD (X'40') for CKD devices
TDCDT1	04	see note below
TDCDT2	05	see note below
TDCDT3	06	see note below
TDCDT4	07	see note below
TDCMSBS	08	Refer to the TDCMSBS default table in appendix A in <i>Maximum Sequential Block Size</i> in the Adabas z/OS installation instructions for more system and device related information.
TDCTPC	0A	Number of tracks per cylinder
TDCCIPT	0C	Number of FBA blocks or PAM pages per track (if TDCFFBA is set). For BS2000 less than or equal to 16.
TDCBPCI	0E	Number of bytes per FBA block or PAM page (2048 if TDCFFBA is set)
TDCABPT	10	Number of Associator blocks per track
TDCABS	12	Associator block size
TDCACPB	14	Number of FBA blocks or PAM pages per Associator block (if TDCFFBA is set)
TDCDBPT	16	Number of Data Storage blocks per track
TDCDBS	18	Data Storage block size
TDCDCPB	1A	Number of FBA blocks or PAM pages per Data Storage block (if TDCFFBA is set)
TDCWBPT	1C	Number of Work blocks per track
TDCWBS	1E	Work block size
TDCWCPB	20	Number of FBA blocks or PAM pages per Work block (if TDCFFBA is set)
TDCTSBPT	22	Number of TEMP or SORT blocks per track (if TDCFFBA is set)
TDCTSBS	24	TEMP or SORT block size
TDCTSCP	26	Number of FBA blocks or PAM pages per TEMP or SORT block (if TDCFFBA is set)
TDCPBPT	28	Number of PLOG blocks per track
TDCPBS	2A	PLOG block size

Label	Offset	Contents
TDCPCPB	2C	Number of FBA blocks or PAM pages per PLOG block (if TDCFFBA is set)
TDCCBPT	2E	Number of CLOG blocks per track
TDCCBS	30	CLOG block size
TDCCCPB	32	Number of FBA blocks or PAM pages per CLOG block (if TDCFFBA is set)

 **Note:** One or more z/VM codes for identifying the device type: the UCB unit type from UCBTBYT4.

In addition, the length of a sequential protection log block may have to be increased. Under z/VM, this length is contained 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 shown below:

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

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

## 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 (block size must be less than or equal to the track size)

### Block Rules for ASSO/DATA

The following rules apply for Associator and Data Storage blocks:

- 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)

### Block Rule for WORK

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

### Block Rules for TEMP/SORT

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

## Block Rule for PLOG or SIBA

The following rules apply for PLOG or SIBA blocks:

- 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 PTF(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 PTF(OUT) then
if QBLKSIZE > 0 then BLKSIZE = QBLKSIZE;
if tape then BLKSIZE = 32760;
else BLKSIZE = TDCMSBS;
else error.

```



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

## Using 3480/3490 Tape Cartridge Compression (IDRC)

The use of hardware compression (IDRC) is not recommended for protection log files. The ADARES BACKOUT function will run much longer when processing compressed data. Also, the BACKOUT function is not supported for compressed data on z/VM systems.





# 6 Installing The AOS Demo Version

---

- AOS Demo Installation Procedure ..... 54
- Installing AOS with Natural Security ..... 54

This section describes how to install the Adabas Online System (AOS) demo version.



**Notes:**

1. To install the full version selectable unit AOS, see the *Adabas Online System* documentation.
2. The AOS demo version requires Natural version 3.1 or above.

## AOS Demo Installation Procedure

---

The Adabas TXTLIB contains the AOS member AOSASM. To install the AOS demo version on a z/VM system, perform the following steps:

▶ **To install the AOS demo version:**

- 1 INPL the AOS demo version dataset

Use the CMS EXEC called ADAINPL provided by Adabas to load the provided AOS demo version INPL tape dataset using Natural Version 3.1 or above.

The dataset itself is in Natural 3 format, and is placed in the library SYSAOS.

- 2 Load the error messages

The error messages are stored in an ERRN-formatted dataset included on the tape.

Use the Natural utility ERRLODUS to load the messages.

See the *Natural Utilities* documentation for information about the ERRLODUS utility.

- 3 Execute the AOS demo version

Log on to the application library SYSAOS and enter the command DBMENU.

## Installing AOS with Natural Security

---

Natural Security must be installed before implementing Adabas Online System Security. See the *Adabas Security* documentation for more information. For information about installing Natural Security for use with AOS Security, see the *Natural Security* documentation.

Natural Security Version 2.2.8 or above includes the ability to automatically close all open databases when the Natural command mode's LOGON function of the AOS demo version is invoked.

Use the following procedure if Natural Security is installed in your environment.

▶ **to setup AOS using Natural Security:**

- 1 Define at least the library SYSAOS to Natural Security

Software AG recommends you define this library and any others you may define as protected.

- 2 Specify the startup program for SYSAOS as DBMENU

Do *not* specify a startup program name for the other libraries.



# 7 Installing The Recovery Aid (ADARAI)

---

- ADARAI Installation Overview ..... 58
- ADARAI Installation Procedure ..... 58

This section describes how to install the Adabas Recovery Aid (ADARAI).

## ADARAI Installation Overview

---

To install the Adabas Recovery Aid, it is necessary to:

- allocate the recovery log;
- customize the skeleton job streams for your installation (see the *Adabas Operations* documentation for more detailed information);
- update the necessary nucleus run/utility job control to include the Recovery Aid data definition statements;
- install the Adabas/ADARAI utility configuration; and
- run ADARAI PREPARE and a save operation to begin a logging generation.

ADARAI is not currently supported by either the ADAMAINT EXEC (database maintenance) or the INSTADA EXEC (database installation). The information needed to run ADARAI must be added manually to the EXECs, CONTROL files, and so on.

## ADARAI Installation Procedure

---

### ► To install the Adabas Recovery Aid:

#### 1 Update ADARAI PARM

Update ADARAI PARM to contain the correct database ID (the default ID is 00001).

#### 2 Allocate the recovery logs

On the same device type, define datasets for the recovery logs DDRLOGR1 and DDRLOGM1.

The DDRLOGR1 and DDRLOGM1 datasets must be CMS-formatted and reserved with the defined file names and file types.

Use the ADAFRM RLOGFRM function to format the RLOGs.

Use the ADAFRM RLOGFRM MIRROR parameter to format the DDRLOGM1 file.

Add MULTI write LINK commands to the ADARAI minidisks in the PROFILE EXEC of the DBA virtual machine.

Define the ADARAI minidisks such as DDRLOGR1 in the CP directory of the Adabas nucleus virtual machine.

### 3 Add data definition statements for the recovery log files

Complete the ADF<sub>dbid</sub> EXEC with DATADEF statements for DDRLOGR1 and, if necessary, DDRLOGM1.

Add these statements to the nucleus job stream and to any utilities that update or save the database and thus write to the RLOG files. Whenever these utilities are executed while ADARAI is active in the database (that is, after the PREPARE function has been executed), the statements must be included.

The following utilities update the database and therefore write to the RLOG:

```
ADAORD (all STORE and REORDER functions)
ADALOD (all functions)
ADAINV (all functions)
ADARES REGENERATE/BACKOUT database
ADASAV RESTORE (all functions) and RESTPLOG
ADADEF NEWWORK
```

The following utilities save the database and therefore write to the RLOG:

```
ADASAV SAVE (all functions)
ADAORD RESTRUCTURE
ADAULD
```

The following utility functions have an impact on recovery and therefore write to the RLOG:

```
ADARES PLCOPY/COPY
ADASAV MERGE
```

Additionally, the Adabas nucleus writes to the RLOG during startup and termination. The nucleus also writes checkpoint information to the RLOG when ADADBS or Adabas Online System functions are processed, ensuring these events are known to ADARAI for recovery processing.

### 4 Install ADARAI on the database.

Execute the ADARAI PREPARE function. ADARAI PREPARE updates the ASSO GCB to indicate that ADARAI is installed. It also creates a control record on the RLOG file with necessary ADARAI information (number of generations, RLOG size, etc.).

### 5 Create the first ADARAI generation.

Execute ADASAV SAVE (database) to start the logging of RLOG information. See the *Adabas Utilities* documentation for more information.

Once ADARAI is active in the database, protection logging must always be used.



# 8

## Managing UES Support of VM Databases

---

- Adding UES Support to an Existing Database ..... 62
- Verifying UES Support ..... 64

You can convert a non-UES-enabled VM database to a UES-enabled one. Once you have done so, you can verify that UES support has been added to a VM database.

## Adding UES Support to an Existing Database

---

► To convert a non-UES-enabled VM database to a UES-enabled one, complete the following steps:

- 1 Load the CMS nucleus extensions required for UES support to your VM database. Use the supplied sample EXEC, NUCXUES, to assist you.



**Note:** The NUCXUES EXEC must be run every time the database is started, so you may find it simpler to call NUCXUES from your ADANUC EXEC.

```

/* ++++++*/ A D A B A S  VERSION 7.4  /+++++ */
/*  NUCXUES  -                               */
/*  This EXEC loads the CMS nucleus extensions required   */
/*  for Adabas UES support.                             */
/*                                                     */
/*  This EXEC is a sample. It is not a part of the ADABAS */
/*  product and is not considered to be supported by any  */
/*  maintenance contract agreements.                     */
/*                                                     */
/*                                                     */
'EXEC DEFNUCX ADAECS'
'EXEC DEFNUCX ADACOX'
'EXEC DEFNUCX SAGSMP2'
'EXEC DEFNUCX SAGECS'
'EXEC DEFNUCX SAGOVO'
return 0
    
```

- 2 Locate the following lines in the ADANUC EXEC for your VM database:

```

address command 'DATADEF DDPRINT,DSN=NUC' || dbid'.DDPRINT,MODE=A'
address command 'DATADEF DDCARD,DSN=ADANUC.DDCARD,MODE=A'
    
```

Once you have located these lines, insert the following lines after them and modify the file modes, as necessary:

```
ADDRESS COMMAND 'DATADEF SYSPARM,DSN=SMARTS.CONFIG,MODE=A'
ADDRESS COMMAND 'DATADEF CONFIG,DSN=CONFIG.RTS,MODE=A'
ADDRESS COMMAND 'DATADEF STDOUT,DSN=STDOUT.DDPRINT,MODE=A'
ADDRESS COMMAND 'DATADEF STDERR,DSN=STDERR.DDPRINT,MODE=A'
```

A sample ADANUC EXEC is provided to assist you.

- 3 Copy the following to a minidisk or SFS directory that is accessible by the database machine.

- APS272.CMSLD01
- BTE421.CMSEC01

Refer to the *Report of Tape Creation* to accurately locate these files.

- 4 Modify SETTXTLB EXEC as follows:

- Update the Global TXTLIB statement putting APSV272 before ADAV744:

```
ADDRESS COMMAND 'GLOBAL TXTLIB APSV272 ADAV744 DMSAMT VMMLIB'
```

- Add a Global LOADLIB statement for APSV272:

```
ADDRESS COMMAND 'GLOBAL LOADLIB APSV272'
```

- 5 Edit the SMARTS CONFIG file provided. Specify a SYSTEM\_ID in SMARTS CONFIG using a value such as the virtual machine name of the VM database. This value is used in messages.

Here is a sample SMARTS CONFIG file:

```
*      SMARTS PARAMETERS
*
CDI=('FILE,PAASFSIO')          NATIVE CMS FILE I/O
SYSTEM_ID=yoursysname
PROCESS_HEAP_SIZE=0
ABEND_RECOVERY=NO
THREAD_ABEND_RECOVERY=NO
LOG=OPER
ASCII=NO
FLOATING_POINT=IEEE
* TRACING PARAMETERS
*SYSTEM_TRACE_LEVEL=5
*TRACE_SYSTEM_INCLUDE=ALL
*TRACE_GROUP_INCLUDE=ALL
```

- 6 Run ADADEF for the VM database, specifying MODIFY UES=YES.

- 7 Start the ITM, the VM database, and Entire Net-Work.

## Verifying UES Support

---

You can verify that UES support has been added to a VM database in one of the following ways:

- When the database starts, it should issue the following message:

```
ADAN7C 00001 ENTIRE CONVERSION SERVICES INITIALIZED
```

- On the Entire Net-Work machine, issue the `DISPLAY TARGETS` command. The display for the UES database should look like this:

```
NET0124I: Target 00001 (I-T) active on node PTGITM
```

The highlighted **T** in this display stands for Translator. If the display shows (I-N), Entire Net-Work does not recognize that the database is UES-enabled.

# 9 Adabas Dump Formatting Tool (ADAFDP)

---

- ADAFDP Function ..... 66
- ADAFDP Output ..... 66

This section describes the use of the Adabas dump formatting tool ADAFDP.

## ADAFDP Function

---

ADAFDP is the address space dump formatting module. During abnormal shutdown of the Adabas nucleus, this module receives control to format and display information that should help you analyze the reason for the error.

During a nucleus shutdown, ADAMPM determines the shutdown reason. If the reason is abnormal termination, ADAMPM loads the ADAFDP module into the address space prior to the 20 call to the Adabas SVC. ADAFDP subsequently receives control to format nucleus information.

If ADAFDP cannot be loaded, message ADAF03 is written to the console and abnormal shutdown continues.

## ADAFDP Output

---

Much of the information formatted by ADAFDP is self-explanatory. However, because the type and amount of information depends on the shutdown situation, a summary of ADAFDP output is provided in this section.

- [ADAFDP Messages](#)
- [Pool Abbreviations](#)
- [User Threads](#)
- [Command Information](#)
- [RABN Information](#)

### ADAFDP Messages

Message	Description
ADAH51 / ADAH52	The message is displayed on the console and written to DDPRINT at the point where the format begins and terminates.
ADAMPM ABEND CODE and PSW	If an abend code and program status word (PSW) were saved in ADAMPM by the Adabas ESTAE, ADAFDP displays these. In addition, ADAFDP determines the module whose entry point best fits the PSW and calculates the offset within that module. If the ADAMPM abend code and PSW are zero, ADAFDP does not format this information.
ADABAS MODULE LOCATIONS	ADAFDP formats and displays the location of each of the Adabas nucleus modules resident in the address space.
ADDRESS LOCATIONS FOR USER EXITS	ADAFDP formats and displays the location of any user exit loaded with the Adabas nucleus.

Message	Description
ADDRESS LOCATIONS FOR HYPEREXITS	ADAFDP formats and displays the location of any hyperexit loaded with the Adabas nucleus. Hyperexits 10-31 are displayed as A-U, respectively.
ADANC0 STANDARD REGISTER SAVE AREA	Registers 0-7/8-F, which are saved in ADANC0. ADAFDP determines if any of these registers contains an address that points at a nucleus pool in storage. If yes, ADAFDP indicates which pool and snaps storage at that address. If the register is 12 and it points to a user thread, ADAFDP snaps the entire thread.
ADANC0 ABEND SAVE REGISTERS	Registers 0-7/8-F, which are saved in ADANC0 as a result of a user abend. ADAFDP determines if any of these saved registers contains an address that points at a nucleus pool in storage. If yes, ADAFDP indicates which pool and snaps storage at that location. If the saved register is 12 and it points to a user thread, ADAFDP snaps the entire thread.
ADAMPM SAVE REGISTERS	Registers 0-7/8-F, which were saved in ADAMPM by the Adabas ESTAE. These are the same registers displayed with the ADAM99 message. ADAFDP determines if any of these saved registers contains an address that points within a nucleus pool in storage. If yes, ADAFDP indicates which pool and snaps storage at that location.
BEGIN / ENDING ADDRESSES OF POOLS / TABLES	ADAFDP determines begin/ending address locations for pools and tables for the Adabas nucleus. These addresses are presented for easy location in the actual dump. See <a href="#">Pool Abbreviations</a> for more information.
ADABAS THREADS	ADAFDP formats the physical threads including threads 0, -1, and -2. The number of lines depends on the value of NT. The thread that was active at the time of the abnormal termination (if any) is marked by a pointer “->”.
USER THREADS	For any of the threads -2 to NT that had assigned work to perform, ADAFDP formats and displays information about the status of that thread. See <a href="#">User Threads</a> for more information:
FOLLOWING COMMANDS WERE FOUND IN THE CMD QUEUE	ADAFDP scans the command queue and formats information for any command found in the queue. See <a href="#">Command Information</a> for more information.
POOL INTEGRITY CHECK	ADAFDP check the integrity of several pools within the Adabas nucleus address space. If an error is detected within that pool, ADAFDP indicates which pool and what type of error was encountered. In addition, ADAFDP snaps storage at the location where the error was detected.
FOLLOWING RABNS / FILES ACTIVE IN BUFFER POOL	ADAFDP scans the buffer pool header for RABNs that were active or being updated. See <a href="#">RABN Information</a> for more information.
ADAIOR REGS FOUND AT OFFSET X'080'	Registers 0-7/8-F found saved in ADAIOR at this offset. If ADAFDP determines that any of these register values is pointing within an Adabas pool, it snaps storage at that location.
ADAIOR REGS FOUND AT OFFSET X'0C0'	Registers 0-7/8-F found saved in ADAIOR at this offset. If ADAFDP determines that any of these register values is pointing within an Adabas pool, it snaps storage at that location.
ICCB POINTED FROM X'A0' IN IOR	The ICCB address to which this offset in ADAIOR points.

Message	Description
ADAI22 ADAIOR TRACE TABLE	Format of ADAIOR trace table; same as that found with the ADAM99 message.

### Pool Abbreviations

Pool Abbreviation	Description
LOG	Log area
OPR	Adabas nucleus operator command processing area
CQ	Address of the command queue, which is formatted later by ADAFDP
ICQ	Internal command queue
TT	Thread table
IA1	Software AG internal area 1
SFT	Session file table
FU	File usage table
FUP	File update table
IOT	I/O table for asynchronous buffer flushing
PL2	PLOG area for asynchronous buffer flushing
PET	Table of posted ETs
TPT	Tpost
TPL	Tplatz
UQP	Unique descriptor pool
UHQ	Upper hold queue
HQ	Hold queue
UUQ	Upper user queue
UQ	User queue
FP	Format pool
FHF	File HILF element
PA	Protection area
TBI	Table of ISNs
TBQ	Table of sequential searches
WK3	Work part 3 space allocation table
IA2	Software AG internal area 2
WK2	Work part 2 space allocation table
VOL	VOLSER table
WIO	Work block I/O area
FST	Free space table work area



Pool Abbreviation	Description
UT	User threads
WP	Work pool
AW2	Work block asynchronous I/O area
IOP	I/O pool related to asynchronous buffer flush
IU2	Buffer pool importance header upper 2
IU1	Buffer pool importance header upper 1
BU2	Buffer pool upper header 2
BU1	Buffer pool upper header 1
BH	Address location of the buffer pool header, information from the buffer pool header is formatted later by ADAFDP
BP	Address location of the physical start of the buffer pool

## User Threads

Information	Description
Thread Number	-2 to NT
Status	Indicates the current status of the thread: <ul style="list-style-type: none"> <li>■ *Active*: the currently active thread</li> <li>■ In Use: thread has been assigned work</li> <li>■ Waiting For I/O: waiting for a block not in buffer pool</li> <li>■ Waiting For RABN: waiting for a RABN already in use</li> <li>■ Waiting For Work-2 Area Block: similar to waiting for I/O</li> <li>■ Waiting Workpool Space: provides number of bytes in decimal</li> <li>■ Ready To Run: waiting to be selected for execution</li> </ul>
CMD	The Adabas command being executed
Response Code	Response code (if any)
File Number	File number for this command
ISN	Internal sequence number for this command
Sub. Rsp	Subroutine response code (if any)
Last RABN for I/O	Last RABN required by command processing, in decimal
Type	Last RABN type (A - ASSO, D - DATA)
CQE Addr	Command queue element address for this command
User Jobname	Job name for user who executed this command
ITID	Internal Adabas ID for user who executed this command
User	User ID for user who executed this command

Information	Description
Unique global ID	28-byte ID for user who owns this command
Buffer Addresses	buffer addresses for: control block, format buffer, search buffer, value buffer, ISN buffer
Buffer Lengths	FL: format buffer length RL: record buffer length SL: search buffer length VL: value buffer length IL: ISN buffer length
Snap Thread	The first 144 bytes of the user thread are snapped

## Command Information

Information	Description
CQE Address	The address location of this CQE
F	<p>Command queue flag bytes:</p> <ul style="list-style-type: none"> <li>■ First Byte: General Purpose Flag                             <ul style="list-style-type: none"> <li>■ X'80': User buffers in service partition, region, address space</li> <li>■ X'40': ET command waiting for 12 call</li> <li>■ X'20': Waiting for 16 call</li> <li>■ X'10': 16 call required</li> <li>■ X'08': Attached buffer</li> <li>■ X'04': Attached buffer required</li> <li>■ X'02': X-memory lock held (MVS only)</li> </ul> </li> <li>■ Second Byte: Selection Flag                             <ul style="list-style-type: none"> <li>■ X'80': In process</li> <li>■ X'40': Ready to be selected</li> <li>■ X'20': Search for UQE done</li> <li>■ X'10': UQE found</li> <li>■ X'08': Not selectable during BSS=x'80' status</li> <li>■ X'04': Not selectable during ET-SYNC</li> <li>■ X'02': Waiting for space</li> <li>■ X'01': Waiting for ISN in HQ</li> </ul> </li> </ul>
CMD	The command type
File Number	The file number for this command
Job Name	Job name for the user
Addr User	UQE Address of users UQE, if searched for and found
Addr User ASCB	Address location of user's ASCB

Information	Description
Addr ECB	Address location of user's ECB (in user's address space)
Addr User UB	Address of users UB (in user's address space)
Addr User PAL	Address location of user's parameter address list
CQE ACA	ACA field of CQE.
CQE RQST	RQST field of CQE
Abuf/Pal	Address of the attached buffer/parameter address list (PAL) for CMD
Comm Id	28-byte unique user ID for this command

### RABN Information

Information	Description
RABN Number	The RABN number in decimal
Type	Type of block (A - ASSO, D - DATA)
Flag	BP header element flag byte: <ul style="list-style-type: none"> <li>■ AKZ X'40': Active indicator</li> <li>■ UKZ X'20': Update indicator</li> <li>■ RKZ X'10': Read indicator</li> <li>■ XKZ X'04': Access is waiting for block</li> <li>■ YKZ X'02': Update is waiting for block</li> <li>■ SKZ X'01': Write indicator</li> </ul>
File	File number that owns this block
Address	Address location of block in storage.



# 10 Online Installation Program (INSTPROD/INSTADA)

---

- Loading INSTADA ..... 74
- Running the Maintenance Version of INSTADA ..... 75
- The INSTADA Display Panels ..... 75

This section describes the INSTADA program for Adabas installation on z/VM systems. INSTADA is invoked internally by the INSTPROD program, which loads product files from installation tapes. INSTADA can also be invoked directly to define new databases. INSTADA must be run at the appropriate step of the z/VM or VM/GCS installation sequence, as described in the section *Install Procedure*.

The installation instructions for Adabas are documented in the section *Install Procedure*. The information in this section presumes that the reader has read and is familiar with that information.



**Note:** INSTADA and ADAMAINT do not support the shared file system (SFS) option or 2-byte DBIDs.

## Loading INSTADA

---

This section describes how to load the INSTADA program.

### ▶ To load the INSTADA program:

- 1 Enter the following commands on the terminal:

```
TAPE FSF 1
```

```
TAPE LOAD * * filemode
```

—where *filemode* is the filemode of the minidisk where the TAPE LOAD command will place the output. After the TAPE LOAD command has completed, the following appears on your terminal:

```
VOL1ADAvrs REST
HDR1ADAvrs.EXEC ADAvrs00010001 88160 000000000000IBM OS/VS 370
HDR2F008000008040DAF0300 /STEP001 B 61918
FILE 'INSTPROD EXEC C1' COPIED.
FILE 'INSTXED1 EXEC C1' COPIED.
R; T=0.03/0.10 15:18:23
```

The first line of the output is the volume serial number of the tape, followed by two lines of header information. The two files INSTPROD EXEC and INSTXED1 EXEC are used to create the online installation panels to unload the product files on the installation tape. These files are copied to the *filemode* minidisk that was specified in the TAPE LOAD command.

- 2 Enter the name of the EXEC that will perform the online installation procedure:

```
INSTPROD
```

This EXEC, in turn, invokes INSTADA to generate the online panels.

## Running the Maintenance Version of INSTADA

For already installed databases, you can run INSTADA to re-define the existing environment by entering the following command in place of INSTPROD:

```
ADAMAINT
```

This command presents the same online installation panel sequence as INSTPROD. However, instead of re-installing the database, ADAMAINT stores the new definition in the library. The new definition becomes effective only after each virtual machine re-accesses the library.

## The INSTADA Display Panels

This section describes the panels that appear when either the INSTPROD or ADAMAINT command is entered.



**Note:** Some panels and fields are not present for VM/GCS systems, as noted in the text. The VM/GCS system is supported by Entire Net-Work, but not by Adabas.

### Preselecting the Database

```
HH:MM:SS          ***** software ag Online Installation *****          YYYY-MM-DD
                   - A D A B A S Installation -                          Panel 0
```


Parameter sets exist for the following Adabas identification numbers

Select a parameter set by marking any ID:

```
    5
   112
```

PF3 will reset modifications

Field	Description
ID (database ID)	To change the parameters of an existing database, enter any character next to the database number. For a new database, press Enter for the next screen.

 **Note:** This panel appears only when databases have already been defined using INSTPROD or ADAMAINT.

### Defining the Environment

This panel is for defining the requirements for the general Adabas environment. If an existing environment was selected on the previous panel, that database name, ID, and present values are displayed in the input fields of this and all following screens. If a new database is being installed, all input fields are empty.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                   - A D A B A S Installation -                               Panel 1

Define your Adabas environment

Modify the following fields to meet your requirements:

    Database name: VMESA-DATABASE112 Database id: 00112 Convert: NO
    Logon IDs of the DB machine: XDB200 ID service machine: DBIDSERV
    ID service machine target ID: 65535 Automatic restart: YES
    System file numbers: 004 000 000 000 000 000 000 000
    Maximum number of files: 020 Demo file numbers: 001 002 003
    Update profile EXECs: YES RR Password for the D-disk: RAD

PF3 will reset modifications
    
```



The following data can be entered on this screen:

Field	Description
Database Name	Name assigned to the database. This name appears in the ADAREP report.
Database ID	Numerical identification to be assigned to the database. A value in the range 1-255 may be specified.
Convert	Change to YES if you are converting an existing database to Adabas.
Logon ID of the DB Machine	Logon of the Adabas virtual machine where the Adabas nucleus will be located.
ID Service Machine	ID of the ID table manager virtual machine. If not specified, the ID is DBIDSERV.
ID Service Machine Target ID	ID of the ID table manager virtual machine.
Automatic Restart (z/VM systems only)	If YES is specified, the ID table manager virtual machine restarts automatically if an abend occurs.
System File Numbers	The first entry is required and must be the checkpoint file. By entering a second number, control statements will be generated for the security file. All additional file numbers are optional.
Maximum Number of Files	Value in the range 5-255 representing the maximum number of files that can be loaded in the database.
Demo File Numbers	File numbers used to load the three demo files: EMPLOYEES, VEHICLES, and MISCELLANEOUS. If any one of these fields is set to zero, the respective demo file will not be loaded.
Update Profile EXECs	If YES is specified, the entries required for Adabas version 7 are added to your existing PROFILE EXEC. If NO is specified, an EXEC named "?????" is generated, which may then be invoked from your PROFILE EXEC.
RR Password for the D-Disk	Enter the read-only password for the Adabas library minidisk where the installation tape is unloaded.


When all correct values have been entered, press `Enter` to make the values effective. If you change values for an existing database and then decide to restore the old values, press `PF3` before pressing `Enter` to restore all old values on the existing panel (other panels are not affected).

After pressing `Enter` to accept entered values, the prompt shown above appears at the bottom of each panel asking you to either confirm by pressing `Enter` again, or to press `PF3` to abort the online installation procedure. In other words, you must press `Enter` twice following entry of each panel data to accept and confirm the entries, and then move to the next panel. If you decide that the values entered on the current panel are not valid, press `PF3` to exit from the procedure and return to the z/VM environment.

On all subsequent screens, pressing `PF3` to exit saves all values entered on previous panels, and restores any preexisting values on the current panel before exiting. If you press `PF3` following the first `Enter`, you must reenter the `INSTPROD` or `ADAMAINT` command to complete the installation. An ID for the partially completed installation appears on the list in the first panel, which you can then select and continue.

### Defining the ID Table Manager

The next panel, shown below, is used to define the environment for the ID table manager virtual machine.

 **Note:** This panel does not appear for VM/GCS systems.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                  - A D A B A S Installation -                               Panel 2

Define your Adabas ID service machine

Modify the following fields to meet your requirements:

    ID service machine target ID: 65535 Automatic restart: YES

    ID service machine: DBIDSERV

    Nodename: ITM65535 DBAVMID: SAGDBA

    Net-work Resource ID: SAGNETWK

PF3 will reset modifications
    
```

Field	Description
ID Service Machine Target ID	Target ID Target ID of the ID table manager virtual machine. This value must be entered.
ID Service Machine	Logon of the ID table manager virtual machine. If you are not using the default DBIDSERV, another name must be entered here.
Nodename	(Entire Net-Work database only). Defines the ID table manager node name for Entire Net-Work.
Automatic Restart	If the ID table manager virtual machine abends, it will restart automatically if YES is specified.
DBAVMID	ID of the database administrator (DBA) virtual machine.
Net-Work Resource ID	Defines the IUCV node name for Entire Net-work.

## Defining the Nucleus Virtual Machine

This panel is the first of a series for defining the Adabas ADARUN parameter values, which define and control the Adabas nucleus.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                    - A D A B A S Installation -                               Panel 3

Define your Adabas nucleus parameters

Modify the following fields to meet your requirements:

    No. of - commands: 020 Held ISNs: 0500 Users: 0020

        Threads: 05 Att. buffs: 016 ISN per TBI: 50

        Held isn per user: 050 TBLES CIDs per user: 05

PF3 will reset modifications
    
```

In the following list, the Field column is the description of the input field on the panel. The ADARUN Parameter column specifies the ADARUN parameter corresponding to the input field. The Default column is the default value in effect if you make no entry. Refer to *Adabas Operations* for specific information about the Adabas ADARUN parameters.

Field	ADARUN Parameter	Default
Commands	NC	200
Held ISNs	NH	500
Users	NU	200
Threads	NT	5
Att. Buffs	NAB	16
ISN per TBI	NSISN	51
Held ISNs per User	NISNHQ	20 (this default is conditional. See <i>Adabas Operations</i> for more information)
TBLES CIDs per User	NQCID	20

This panel continues the definition of the Adabas ADARUN parameter values for z/VM installation.



**Note:** This panel does not appear in the VM/GCS sequence.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                    - A D A B A S Installation -                               Panel 4

Define your Adabas nucleus parameters

Modify the following fields to meet your requirements:

    Size of - buffer pool: 0250000 FB pool: 012000 ISN lists: 010000

    Prot. area: 01000 Work pool: 150000 Sort area: 050000

    User buffer: 065535 Table of sequential commands: 010000

    UQ Pool: 0005000 Asynch. Buffer flush Pool: 0000000

    Work Part 2: 00000

PF3 will reset modifications
    
```

In the following list, the Field column is the description of the input field on the panel. The ADARUN Parameter column specifies the ADARUN parameter corresponding to the input field. The Default column is the default value in effect if you make no entry. Refer to *Adabas Operations* for specific information about the Adabas ADARUN parameters.

Field	ADARUN Parameter	Default
Buffer Pool	LBP	250000
FB Pool	LFP	12000
ISN Lists	LI	10000
Protection Area	LP	10000
Work Pool	LWP	150000
Sort Area	LS	49920
User Buffer	LU	65535
Table of Sequential Commands	LQ	10000
UQ Pool	LDEUQP	5000
Asynchr. Buffer Flush Pool	LFIOP	0/6000 (the default is zero to disable asynchronous buffer flushing, or a minimum of 6000 to enable asynchronous buffer flushing. See <i>Adabas Operations</i> for more information)
Work, Part 2	LWKP2	0

This panel continues the definition of the Adabas ADARUN parameter values, and applies to both z/VM and VM/GCS systems.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD

                      - A D A B A S Installation -                               Panel 5

Define your Adabas nucleus parameters

Modify the following fields to meet your requirements:

    Read-only: NO Utility-only: NO Open command required: NO

    Time windows - command: 0060 Search: 0900 Transaction: 0900

    Non-activity - access: 0900 ET/BT: 0900 Exclusive: 0900

                      - Online services: 0900

    Sequential block size: 00000          Buffer Flush Duration: 001

PF3 will reset modifications
    
```

In the following list, the Field column is the description of the input field on the panel. The ADARUN Parameter column specifies the ADARUN parameter corresponding to the input field. The Default column is the default value in effect if you make no entry. Refer to *Adabas Operations* for specific information about the Adabas ADARUN parameters.

Field	ADARUN Parameter	Default
Read-only	READONLY	NO
Utility-only	UTIONLY	NO
OPEN command required	OPENRQ	YES
Time windows-command	CT	60 (see note below)
Search	TLSCMD	300 (see note below)
Transaction	TT	900 (see note below)
Non-activity-access	TNAA	900 (see note below)
ET/BT	TNAE	900 (see note below)
Exclusive Use	TNAX	900 (see note below)
Sequential block size	QBLKSIZE	0
Buffer Flush Duration	TFLUSH	1 (see note below)



**Note:** All default or specified times are in units of 1.08 seconds. See *Adabas Operations* for more information.

### Defining Protection Logging

This panel accepts information associated with protection logging.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD

                - A D A B A S Installation -                               Panel 4

Define your Adabas nucleus parameters (Protection logging)
Modify the following fields to meet your requirements:

    Protection logging: NO Protection logging required: NO

    Dual logging device: 3380 Size: 000120 CMS-directory: YES

    Dual logging file name: TESTCMS.PLOG


    Unit 1: 440 Volume: DPLOG1 Link Unit: 440 Password: MAD

    Unit 2: 441 Volume: DPLOG2 Link Unit: 441 Password: MAD

PF3 will reset modifications
    
```

Field	Description
Protection Logging	Enter either YES or NO. Even if you specify NO, first read and provide the applicable information for other panel fields before continuing to the next panel.  If NO is entered, then one of the following cases applies: <ul style="list-style-type: none"> <li>■ With a DUALPLS parameter, the PLOG minidisk is used;</li> <li>■ With the DATADEF parameter for SIBA, the SIBA minidisk is used;</li> <li>■ With neither parameter, no protection logging occurs.</li> </ul>
Protection Logging Required	If YES is specified, the ADARUN parameter PLOGRQ=YES is created. In this case, any attempt to start an Adabas nucleus without a protection log causes Adabas initialization to terminate with an error message.
Dual Logging Device	Enter the device type to be used. (3380, for example).
Size	Enter the number of blocks available for each PLOG minidisk. See <i>Device and File Considerations</i> , elsewhere in this guide, for blocksize information.

Field	Description
CMS-Directory	Enter YES if the dual PLOG is a z/VM minidisk; enter NO if the dual PLOG is an OS-formatted disk.
Dual Logging File Name	Enter the file name and file type of the file.
Unit 1 and Unit 2	Enter the CUU of the minidisk defined to the Adabas virtual machine for the protection log.
Volume	Enter the volume name, (i.e., LABEL) of each minidisk identified.
Link Unit	Enter the link CUU for each PLOG minidisk identified.
Password	Enter the MULTI-WRITE password for each PLOG minidisk identified.

 **Caution:** Failure to enter the MW link passwords when implementing dual protection logging will cause the installation to terminate abnormally since no CP LINK statements will be generated.

### Defining Command Logging

This panel accepts information that controls command logging.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                    - A D A B A S Installation -                               Panel 5

Define your Adabas nucleus parameters (Command logging)
Modify the following fields to meet your requirements:

    Command logging: NO Command log size: 04000

    Dual logging device: 3380 Size: 000135 CMS-directory: YES

    Dual logging file name: TESTCMS.CLOG

    Unit 1: 430 Volume: DCLOG1 Link Unit: 430 Password: MAD

    Unit 2: 431 Volume: DCLOG2 Link Unit: 431 Password: MAD

    Log - CB: NO FB: NO RB: NO SB: NO VB: NO IB: NO IO: NO

PF3 will reset modifications
    
```

Field	Description
Command Logging	Enter YES or NO. If you enter NO, you can continue directly to the next panel.
Command Log Size	Enter the blocksize if using DDLOG (single logging).
Dual Logging Device	Enter the device type to be used. (3380, for example).
Size	Enter the number of blocks available for each CLOG minidisk. See <i>Device and File Considerations</i> , elsewhere in this guide, for blocksize information.
CMS-Directory	Enter YES if the dual CLOG is a z/VM minidisk; enter NO if the dual CLOG is an MVS-formatted disk.
Dual Logging File Name	Enter the file name and file type of the file.
Unit 1 and Unit 2	Enter the CUU of the minidisk defined to the Adabas virtual machine for the command log.
Volume	Enter the volume name, (i.e., LABEL) of each minidisk identified.
Link Unit	Enter the link CUU for each CLOG minidisk identified.
Password	Enter the MULTI WRITE password for each CLOG minidisk identified.
Log	Select command logging controls; valid only if command logging on this panel specifies YES. Enter YES or NO to log the following information: <ul style="list-style-type: none"> <li>■ Log CB: the Adabas control block</li> <li>■ Log FB: the format buffer</li> <li>■ Log RB: the record buffer</li> <li>■ Log SB: the search buffer</li> <li>■ Log VB: the value buffer</li> <li>■ Log IB: the ISN buffer</li> <li>■ Log IO: the I/O activity</li> </ul>

### Defining the Adabas Associator

This panel is used to define the Adabas Associator.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                  - A D A B A S Installation -                               Panel 7

Define your ASSOCIATOR structure for FILE 1

Device type: 3380 Size in cylinders: 00049 CMS-directory: YES

File name: TESTCMS.ASS01

Volume labels: ASS200

Unit addresses: 200 000 000 000
    
```



```

Link addresses: 200 000 000 000

Link passwords: MAD

More volumes for file: NO More ASSO files: NO
    
```

PF3 will reset modifications

Field	Description
Device Type	Enter the device type to be used.
Size in Cylinders	Enter the size of minidisks as the number of cylinders in the MDISK statement minus 1. A warning will be displayed, reminding you that Adabas does not use the first cylinder of the minidisk.
CMS-Directory	Enter YES if ASSO is on a z/VM minidisk; enter NO if ASSO is on an MVS-formatted disk.
File Name	Enter the file name and the file type of the file.
Volume Label(s)	Enter up to four volume names for ASSO.
Unit Address(es)	Enter the corresponding CUUs for the volumes.
Link Address(es)	Enter the link CUUs being used for ASSO.
Link Password(s)	Enter the corresponding MULTI WRITE passwords for each volume.
More Volumes for File	Enter YES if you want to define more than four volumes for ASSO. Another panel will be provided for additional volumes.
More ASSO Files	YES produces panel to define ASSOR2, 3, 4, and 5.



**Note:** Adabas does not use the first cylinder of a z/VM-formatted disk. Therefore, when specifying size, use the actual size in number of cylinders, minus 1. Do this for every minidisk used for Adabas files. In addition, subtract one cylinder for every additional volume specified.

### Defining the Adabas Data Storage

This panel is used to define the Adabas Data Storage.

```

HH:MM:SS                ***** software ag Online Installation *****
YYYY-MM-DD                - A D A B A S Installation -                               Panel 8

Define your DATA STORAGE structure for FILE 1

Device type: 3380 Size in cylinders: 00111 CMS-directory: YES
    
```

```

File name: TESTCMS.DATA1

Volume labels: DAT300 DAT301 DAT302 DAT303

Unit addresses: 300 301 302 303

Link addresses: 300 301 302 303

Link passwords: MAD MAD MAD MAD

More volumes for file: NO More DATA files: NO

PF3 will reset modifications
    
```

Field	Description
Device Type	Enter the device type to be used (3380, for example).
Size in Cylinders	Enter the size of the minidisk in number of cylinders, minus 1.
CMS-Directory	Enter YES if DATA is on a z/VM minidisk; enter NO if DATA is on an MVS-formatted disk.
File Name	Enter the file name and file type of the file.
Volume Label(s)	Enter up to four volume names for DATA.
Unit Address(es)	Enter the corresponding CUUs for the volumes.
Link Address(es)	Enter the link CUUs that will be used on this virtual machine.
Link Password(s)	Enter the corresponding MULTI WRITE passwords for each volume.
More Volumes for File	Enter YES if you want to define more than four volumes for DATA. Another panel will be provided for additional volumes.
More DATA files	Enter YES to produce panel to define DATAR2, 3, 4, or 5.

**Defining the Adabas Work Area**

This panel is used to define the Adabas Work area.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD

                      - A D A B A S Installation -                               Panel 9

Define your WORK STORAGE structure for FILE 1

Device type: 3380 Size in cylinders: 00024 CMS-directory: YES

File name: TESTCMS.WORK1
    
```

```

Volume labels: WOR400

Unit addresses: 400 000 000 000

Link addresses: 400 000 000 000

Link passwords: MAD

More volumes for file: NO More WORK files: NO
    
```

PF3 will reset modifications

Field	Description
Device Type	Enter the device type to be used (3380, for example).
Size in Cylinders	Enter the size of the minidisk in number of cylinders, minus 1.
CMS-Directory	Enter YES if WORK is on a z/VM minidisk; enter NO if WORK is on an MVS-formatted disk.
File Name	Enter the file name and file type of the file.
Volume Label(s)	Enter up to four volume names for WORK.
Unit Address(es)	Enter the corresponding CUUs for the volumes.
Link Address(es)	Enter the link CUUs that will be used on this virtual machine.
Link Password(s)	Enter the corresponding MULTI WRITE passwords for each volume.
More Volumes for File	Enter YES if you want to define more than four volumes for WORK. Another panel will be provided for additional volumes.

### Defining the Adabas Temp Area

This panel is used to define the Adabas Temp area.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                   - A D A B A S Installation -                               Panel
10

Define your TEMPORARY STORAGE structure for FILE 1

Device type: 3380 Size in cylinders: 00009 CMS-directory: YES

File name: TESTCMS.TEMP1

Volume labels: TEM410

Unit addresses: 410 000 000 000
    
```

```

Link addresses: 410 000 000 000

Link passwords: MAD

More volumes for file: NO More TEMP files: NO

PF3 will reset modifications
    
```

Field	Description
Device Type	Enter the device type to be used (3380, for example).
Size in Cylinders	Enter the size of the minidisk in number of cylinders, minus 1.
CMS-Directory	Enter YES if TEMP is on a z/VM minidisk; enter NO if TEMP is on an MVS-formatted disk.
File Name	Enter the file name and file type of the file.
Volume Label(s)	Enter up to four volume names for WORK.
Unit Address(es)	Enter the corresponding CUUs for the volumes.
Link Address(es)	Enter the link CUUs that will be used on this virtual machine.
Link Password(s)	Enter the corresponding MULTI WRITE passwords for each volume.
More Volumes for File	Enter YES if you want to define more than four volumes for TEMP. Another panel will be provided for additional volumes.

### Defining the Adabas Sort Area

This panel is used to define the Adabas Sort area.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                   - A D A B A S Installation -                               Panel
11

Define your SORT STORAGE structure for FILE 1

Device type: 3380 Size in cylinders: 00009 CMS-directory: YES

File name: TESTCMS.SORT1

Volume labels: SOR420

Unit addresses: 420 000 000 000

Link addresses: 420 000 000 000
    
```

Link passwords: MAD

More volumes for file: NO More SORT files: YES

PF3 will reset modifications

Field	Description
Device Type	Enter the device type to be used (3380, for example).
Size in Cylinders	Enter the size of the minidisk in number of cylinders, minus 1.
CMS-Directory	Enter YES if SORT is on a z/VM minidisk; enter NO if SORT is on an MVS-formatted disk.
File Name	Enter the file name and file type of the file.
Volume Label(s)	Enter up to four volume names for WORK.
Unit Address(es)	Enter the corresponding CUUs for the volumes.
Link Address(es)	Enter the link CUUs that will be used on this virtual machine.
Link Password(s)	Enter the corresponding MULTI WRITE passwords for each volume.
More Volumes for File	Enter YES if you want to define more than four volumes for SORT. Another panel will be provided for additional volumes.
More SORT Files	YES produces a panel to define SORT2.

### Defining the Checkpoint/ET File

This panel accepts ADALOD utility parameters used for loading the checkpoint file.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                  - A D A B A S Installation -                               Panel
12

Define your Checkpoint and ET-data File

Modify the following fields to meet your requirements:

File name: CHECKPOINT Max. ISN: 001000 Min. CP ISN: 00255

ASSO padding factor: 10 AC-RABN: 00000

Normal index size: 64B NI-RABN: 00000 Max. secondary alloc: 00000

Upper index size: 03B UI-RABN: 00000 Max. secondary alloc: 00000

DATA padding factor: 10 DS-RABN: 00000 Max. secondary alloc: 00000

```

DS-reusage: NO DS-size: 100B DS-device: 3380

PF3 will reset modifications

The left column equates to the description prior to the input field on the panel. The right column represents the ADALOD parameter associated with the description. Refer to *Adabas Utilities* for more specific information on the ADALOD parameters.

Field	ADARUN Parameter
File Name	NAME
Max ISN	MAXISN
Min CP ISN	MINISN
ASSO Padding Factor	ASSOPFAC
AC-RABN	ACRABN
Normal Index Size	NISIZE
NI-RABN	NIRABN
Max Secondary Alloc	MAXNI
Upper Index Size	UISIZE
UI-RABN	UIRABN
Max Secondary Alloc	MAXUI
DATA Padding Factor	DATAPFAC
DS-RABN	DSRABN
Max Secondary Alloc	MAXDS
DS-Reusage	DSREUSE
DS-Size	DSSIZE
DS-Device	DSDEV

### Demo File Definition (EMPLOYEES)

This panel accepts ADALOD parameters used when loading the EMPLOYEES demo file.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                  - A D A B A S Installation -                               Panel
14

Define your DEMO-File 1, Employees File

Modify the following fields to meet your requirements:
    
```

```

File name: EMPLOYEES Max. ISN: 001500

ASSO padding factor: 10 AC-RABN: 00000

Normal index size: 140B NI-RABN: 00000 Max. secondary alloc: 00000

Upper index size: 15B UI-RABN: 00000 Max. secondary alloc: 00000

DATA padding factor: 10 DS-RABN: 00000 Max. secondary alloc: 00000

DS-reusage: YES DS-size: 110B DS-device: 3380
    
```

PF3 will reset modifications

Field	ADARUN Parameter
File Name	NAME
Max ISN	MAXISN
Min CP ISN	MINISN
ASSO Padding Factor	ASSOPFAC
AC-RABN	ACRABN
Normal Index Size	NISIZE
NI-RABN	NIRABN
Max Secondary Alloc	MAXNI
Upper Index Size	UISIZE
UI-RABN	UIRABN
Max Secondary Alloc	MAXUI
DATA Padding Factor	DATAPFAC
DS-RABN	DSRABN
Max Secondary Alloc	MAXDS
DS-Reusage	DSREUSE
DS-Size	DSSIZE
DS-Device	DSDEV

### Demo File Definition (VEHICLES)

This panel accepts ADALOD parameters when loading the VEHICLES demo file.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                                - A D A B A S Installation -                               Panel
15

Define your DEMO-File 2, Vehicles File

Modify the following fields to meet your requirements:

    File name: VEHICLES Max. ISN: 001500

    ASSO padding factor: 10 AC-RABN: 00000

    Normal index size: 35B NI-RABN: 00000 Max. secondary alloc: 00000

    Upper index size: 10B UI-RABN: 00000 Max. secondary alloc: 00000

    DATA padding factor: 10 DS-RABN: 00000 Max. secondary alloc: 00000

    DS-reusage: YES DS-size: 50B DS-device: 3380

PF3 will reset modifications
    
```

Field	ADARUN Parameter
File Name	NAME
Max ISN	MAXISN
Min CP ISN	MINISN
ASSO Padding Factor	ASSOPFAC
AC-RABN	ACRABN
Normal Index Size	NISIZE
NI-RABN	NIRABN
Max Secondary Alloc	MAXNI
Upper Index Size	UISIZE
UI-RABN	UIRABN
Max Secondary Alloc	MAXUI
DATA Padding Factor	DATAPFAC
DS-RABN	DSRABN
Max Secondary Alloc	MAXDS



Field	ADARUN Parameter
DS-Reusage	DSREUSE
DS-Size	DSSIZE
DS-Device	DSDEV

### Demo File Definition (MISCELLANEOUS)

This panel accepts ADALOD parameters to load the MISCELLANEOUS demo file.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                                - A D A B A S Installation -
16                                                                Panel

Define your DEMO-File 3, MISCELLANEOUS File

Modify the following fields to meet your requirements:

    File name: MISCELLANEOUS Max. ISN: 002000

    ASSO padding factor: 10 AC-RABN: 00000

    Normal index size: 50B NI-RABN: 00000 Max. secondary alloc: 00000

    Upper index size: 07B UI-RABN: 00000 Max. secondary alloc: 00000

    DATA padding factor: 10 DS-RABN: 00000 Max. secondary alloc: 00000

    DS-reusage: YES DS-size: 76B DS-device: 3380

PF3 will reset modifications
    
```

Field	ADARUN Parameter
File Name	NAME
Max ISN	MAXISN
Min CP ISN	MINISN
ASSO Padding Factor	ASSOPFAC
AC-RABN	ACRABN
Normal Index Size	NISIZE
NI-RABN	NIRABN
Max Secondary Alloc	MAXNI
Upper Index Size	UISIZE

Field	ADARUN Parameter
UI-RABN	UIRABN
Max Secondary Alloc	MAXUI
DATA Padding Factor	DATAPFAC
DS-RABN	DSRABN
Max Secondary Alloc	MAXDS
DS-Reusage	DSREUSE
DS-Size	DSSIZE
DS-Device	DSDEV

### Installing Adabas

After panel 16 has been completed, the remaining online panels require no input. They simply display statements giving the status of the installation procedure.

```

HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                  - A D A B A S Installation -                               Panel
18

The Adabas execution environment for data base 112 is being established

The PROFILE EXEC for the ID table service machine already exists

The zap file for the ID table service machine already exists

The zap file for ADALDI already exists

The PROFILE EXEC XDB200 has been created

The ADAFRM control cards to format DB 112 already exist

The PROFILE EXEC for SAGDBA has been created or modified

The ADADEF control cards to define DB 112 already exist

Hit ENTER to proceed
    
```

During the building of panel 18, messages will appear giving a step by step status of the installation process. During the building of panel 18 there will be two interruptions, and at both times the literal "MORE..." will be displayed on the bottom right corner of the terminal. The messages that will appear will be the names of the ID table service machine and the Adabas nucleus machine. Output should look similar to the following:

```
DBIDSERV: R; T=0.01/0.01 15:33:20
ADABASV: R; T=0.01/0.01 15:33:20
```

This message will indicate that a PROFILE EXEC has been created for these machines and has been punched over to that virtual machine. It will be necessary to press the `Clear` key to continue the installation process.

When panel 18 is completed, you will be prompted to press the `Enter` key to proceed.

```
HH:MM:SS          ***** software ag Online Installation *****
YYYY-MM-DD
                    - A D A B A S Installation -                               Panel
19
The Adabas execution environment for database 112 is being established
The control cards to run the nucleus for DB 112 already exist
The ADARUN control cards for device 3380 already exist
The control cards to load demo-file EMPLOYEES already exist
The control cards to load demo-file VEHICLES already exist
The control cards to load demo-file MISCELLANEOUS already exist
The volume list for DB 112 has been created
The ADADEFS EXEC for DB 112 has been created
Hit ENTER to proceed
```

Panel 19 will continue to display updates on the creation of ADARUN control cards and the control cards to load the demonstration files. At the end of all the messages, you will be prompted to press `Enter` to proceed. At this point, the database environment has been set up. If the procedure is not to be continued, press `PF3` to terminate.

LINK statements should appear for each Adabas file required for the installation followed by a message indicating that the Adabas library minidisk is being accessed as READ ONLY. The output should look similar to the following:

```
DASD 200 LINKED R/W; R/W BY ADABASVv
DASD 300 LINKED R/W; R/W BY ADABASVv
DASD 400 LINKED R/W; R/W BY ADABASVv
DASD 410 LINKED R/W; R/W BY ADABASVv
DASD 420 LINKED R/W; R/W BY ADABASVv
DMSACC724I '202' REPLACES ' C (202) '
DMSACC723I C (202) R/O
```

If a LINK statement is *not* displayed for each required Adabas file, ADAFRM will terminate abnormally. Check to see that the correct MULTI WRITE passwords were entered for each Adabas file.

The installation procedure can be restarted to change passwords or make other corrections by entering:

```
INSTADA
```

The installation procedure continues by invoking the following EXECs to perform the previously mentioned functions:

```
DBINIT perform z/VM minidisk functions
ADAFRM format the files for the database
ADADEF define the database
ADALOD load the three demo files
ADAREP produce a database report
```

```
The Adabas environment for database 112 has been successfully modified
Ready;
```

```
RUNNING HOSTSYS
```

The final step of the installation procedure is to bring the nucleus up if the Adabas machine has the DBA machine defined as a secondary console. The following message appears after that step has started:

```
The Adabas nucleus for database 225 has been started
Adabas installed successfully
READY;
```

Whether you are on the DBA or Adabas machine, the following messages are displayed:

```
ADAN02 00225 NUCLEUS-RUN WITHOUT PROTECTION-LOG
ADAN03 00225 ADABAS COMING UP
ADAN01 00225 A D A B A S IS ACTIVE
ADAN01 00225 MODE = MULTI
```

The installation has finished, and the nucleus should be active.



# 11 DATADEF Information For VM/GCS

---

- DATADEF File Assignments ..... 100
- DATADEF Error Codes ..... 102

The information in this section is applicable for Entire Net-Work systems that run in a z/VM / GCS environment.

## DATADEF File Assignments

All assignments for files which are accessed by an Adabas nucleus or utility must be done using DATADEF which replaces the GCS FILEDEF command in the Adabas environment. DATADEF accepts parameters in the form of either a tokenized parameter list or an extended parameter list. The extended parameter list takes precedence. The file assignments established by the DATADEF statements can be listed using the DISPDD program.

The program RELDD can be used to clear active DATADEF entries. RELDD accepts a list of file names to be released or, if no list is specified, clears all active DATADEF entries.

The DATADEF statement creates a data definition block (DDB), which remains in system storage until it is either

- overwritten by another DATADEF statement with the same name;
- cleared by RELDD;
- or cleared by a GCS IPL.

The parameters for DATADEF consist of one positional parameter and one or more keyword parameters separated by commas. An equal sign (=) must be used between a keyword and the parameter value. Depending on how DATADEF is invoked, spaces may be required surrounding equal signs, commas, and parentheses.

The DATADEF parameters are described in the following table:

Parameter Keyword	Required/ Optional	Maximum Length	Specifies . . .
positional	Required	8	the file name (DD) names as specified in <i>Adabas Operations</i> or <i>Adabas Utilities</i> .
BUFNO	Optional	3 (1 - 255)	the number of buffers to be allocated for a sequential file on tape. Default: 3
COMPRESS	Optional	53 (YES - NO)	whether or not an output file on tape should make use of IDRC available on certain cassette units. If IDRC is not supported, this parameter is set to NO, the default.
CONCAT	Optional	5 (1 - 255)	a concatenation sequence number for the file. This results in the DDB being concatenated to another existing DDB with the same file name. The sequence numbers must be specified in ascending order with no numbers left out. The first file to be concatenated has the number 1. If specified for a file to be read backward, the sequence numbers are to



Parameter Keyword	Required/ Optional	Maximum Length	Specifies . . .
			be given in the normal sequential order. A DATADEF statement without CONCAT frees any existing root DDB and any DDBs concatenated to it.
DISP	Optional for output only	3	whether a sequential file is to be created (NEW), extended (MOD) or overwritten (OLD). If NEW is specified and the file exists a return code is issued. If OLD is specified, the file's existence is not checked. Default: OLD
DSN	Required if not dummy	44	the dataset name.
DUMMY	Optional	-	that the file does not exist. DUMMY is a keyword without a value. It may not be specified with any other parameter except file name.
EXTEND	Optional	-	that an existing DDB is to be extended. EXTEND is a keyword without a value. It may only be specified with the file name, VOL, and UNIT parameters.
FILESEQ	Tape only, optional	3	the sequence number of the file on a multi-file tape. The default value (1) is required if tapes are to be read backward.
FNAME	Required if DSN is a SFS	8	the file name of an SFS member.
FTYPE	Required if DSN is a SFS	8	the file type of an SFS member.
LRECL	See note 2	5	the length of the physical blocks in the file. If RECFM=FB, BLKSIZE must be an integral multiple of LRECL; if RECFM=V or RECFM=VB, BLKSIZE must be equal to or more than LRECL + 4.
MODE	See note 1	2	the z/VM filemode.
RECFM	See note 2	2	the format of the records in the file (F, FB, V, VB, U).
UNLOAD	Tape only, optional	3 (YES - NO)	whether or not the tape is rewound and unloaded. If NO is specified, the tape is rewound at close but is not unloaded. Default: YES.
UNIT	See notes 1 and 3	See note 1	a list of virtual addresses ( <i>cuu</i> , or <i>ccuu</i> for XA) of the unit or units containing the file, or one of the logical device abbreviations: TRM, PUN, RDR, PRT, or SFS. If a unit address list is given, it must be enclosed in parentheses and entries must be separated by commas.
VOL	See note 1	See note 1	a list of the serial numbers (each at most 6 characters) of the volumes containing the file; if multiple volumes are specified, they must be separated by commas and enclosed in parentheses.

Notes:

1. A `MODE` parameter is required for sequential z/VM DASD files. For DASD volumes containing database files, either a `VOL`, `UNIT`, or `MODE` parameter is required. If the database file spans multiple volumes, `VOL` or `UNIT` must be specified. Specifying `MODE=*` for a non-existent file results in a return code of 32.

If both `VOL` and `UNIT` are specified, the number of volumes and unit addresses must be equal and each volume in the `VOL` list must be mounted on the unit specified by the corresponding entry in the `UNIT` list (the first `VOL` entry must be mounted on the first `UNIT` entry, and so on).

For tape files, a `UNIT` must be specified and a real unit attached prior to `DATADEF` execution, unless the tape unit is dynamically allocated at open time; in this case, `TAPx` can be specified according to standard z/VM conventions, where `TAP1` specifies virtual unit 181, `TAP2` specifies unit 182, and so on. Only one tape unit address is allowed in the `UNIT` parameter.

The `VOL` parameter is required for input tape files, but is optional for output tape files. If a tape file spanning multiple volumes is to be read backwards, specify the volumes in the normal sequential order.

When creating a multi-volume tape file, Adabas-z/VM maintains a list of the file volumes. To refer to that volume list in a later `DATADEF`, specify `VOL=*filename` where `filename` is the name of the multi-volume file.

2. The parameters `RECFM`, `LRECL`, and `BLKSIZE` are required only for tape input files without a `HDR2` label and for `VSE` sequential DASD files. If `RECFM` has been specified, the corresponding `BLKSIZE` and `LRECL` parameters are also required.

## DATADEF Error Codes

The following error codes may be returned by `DATADEF`:

Response Code	Description
16	No parameter list was supplied
20	Invalid keyword
24	No file name specified
28	Error in <code>DSN</code> or <code>DUMMY</code> specification: neither a <code>DSN</code> nor <code>DUMMY</code> was specified; or conflicting parameter specification for a dummy file
32	Error in <code>VOL</code> , <code>UNIT</code> or <code>MODE</code> parameter
36	Incorrect length for <code>VOL</code> , <code>UNIT</code> or <code>MODE</code> parameter
40	Insufficient virtual storage
44	Internal error issuing a <code>CP</code> command
48	Invalid <code>cuu</code> address (internal error)
52	Volume or unit not available or non-VTOC volume has been attached

Response Code	Description
56	Database file resides on volumes with mixed formats (VTOC, non-VTOC)
60	Database file resides on volumes of different device types
64	More than one unit specified for a tape file
68	Invalid file sequence number
72	Invalid RECFM parameter
76	Invalid BLKSIZE parameter
80	Invalid LRECL parameter
84	Invalid DISP parameter
88	Invalid concatenation count
92	Invalid DDB extension



# 12 Translation Tables

---

- Adabas EBCDIC to ASCII and ASCII to EBCDIC ..... 106
- Entire Net-Work EBCDIC to ASCII and ASCII to EBCDIC ..... 107

This section describes the translation tables which are supplied by Adabas.

## Adabas EBCDIC to ASCII and ASCII to EBCDIC

```

cUES2ASC DS 0F
c* .0.1.2.3.4.5.6.7.8.9.A.B.C.D.E.F
c DC x'000102033F093F7F3F3F3F0B0C0D0E0F' 0.
c DC x'101112133F3F083F18193F3F3F1D3F1F' 1.
c DC x'3F3F1C3F3F0A171B3F3F3F3F050607' 2.
c DC x'3F3F163F3F1E3F043F3F3F3F14153F1A' 3.
c DC x'203F3F3F3F3F3F3F3F3F2E3C282B3F' 4.
c DC x'263F3F3F3F3F3F3F3F3F21242A293B5E' 5.
c DC x'2D2F3F3F3F3F3F3F3F3F7C2C255F3E3F' 6.
c DC x'3F3F3F3F3F3F3F3F3F603A2340273D22' 7.
c DC x'3F6162636465666768693F3F3F3F3F' 8.
c DC x'3F6A6B6C6D6E6F7071723F3F3F3F3F' 9.
c DC x'3FE7E37475767778797A3F3F3F5B3F3F' A.
c DC x'3F3F3F3F3F3F3F3F3F3F3F3F5D3F3F' B.
c DC x'7B4142434445464748493F3F3F3F3F' C.
c DC x'7D4A4B4C4D4E4F5051523F3F3F3F3F' D.
c DC x'5C3F535455565758595A3F3F3F3F3F' E.
c DC x'303132333435363738393F3F3F3F3F' F.
c* .0.1.2.3.4.5.6.7.8.9.A.B.C.D.E.F
END
cUES2EBC DS 0F
c* .0.1.2.3.4.5.6.7.8.9.A.B.C.D.E.F
c DC x'00010203372D2E2F1605250B0C0D0E0F' 0.
c DC x'101112133C3D322618193F27221D351F' 1.
c DC x'405A7F7B5B6C507D4D5D5C4E6B604B61' 2.
c DC x'F0F1F2F3F4F5F6F7F8F97A5E4C7E6E6F' 3.
c DC x'7CC1C2C3C4C5C6C7C8C9D1D2D3D4D5D6' 4.
c DC x'D7D8D9E2E3E4E5E6E7E8E9ADE0BD5F6D' 5.
c DC x'79818283848586878889919293949596' 6.
c DC x'979899A2A3A4A5A6A7A8A9C06AD0A107' 7.
c DC x'6F6F6F6F6F6F6F6F6F6F6F6F6F6F' 8.
c DC x'6F6F6F6F6F6F6F6F6F6F6F6F6F6F' 9.
c DC x'6F6F6F6F6F6F6F6F6F6F6F6F6F6F' A.
c DC x'6F6F6F6F6F6F6F6F6F6F6F6F6F6F' B.
c DC x'6F6F6F6F6F6F6F6F6F6F6F6F6F6F' C.
c DC x'6F6F6F6F6F6F6F6F6F6F6F6F6F6F' D.
c DC x'6F6F6F6F6F6F6F6F6F6F6F6F6F6F' E.
c DC x'6F6F6F6F6F6F6F6F6F6F6F6F6F6F' F.
c* .0.1.2.3.4.5.6.7.8.9.A.B.C.D.E.F
END

```

## Entire Net-Work EBCDIC to ASCII and ASCII to EBCDIC

```

NW2ASC DS OF
* .0.1.2.3.4.5.6.7.8.9.A.B.C.D.E.F
DC X'000102030405060708090A0B0C0D0E0F' 0.
DC X'101112131415161718191A1B1C1D1E1F' 1.
DC X'00000000000000000000000000000000' 2.
DC X'00000000000000000000000000000000' 3.
DC X'20000000000000000000000005B2E3C282B5D' 4.
DC X'260000000000000000000000021242A293B5E' 5.
DC X'2D2F000000000000000000007C2C255F3E3F' 6.
DC X'00000000000000000000000603A2340273D22' 7.
DC X'0061626364656667686900000000000000' 8.
DC X'006A6B6C6D6E6F70717200000000000000' 9.
DC X'007E737475767778797A000005B000000' A.
DC X'000000000000000000000000000005D0000' B.
DC X'7B41424344454647484900000000000000' C.
DC X'7D4A4B4C4D4E4F50515200000000000000' D.
DC X'5C7E535455565758595A00000000000000' E.
DC X'303132333435363738397C00000000FF' F.
* .0.1.2.3.4.5.6.7.8.9.A.B.C.D.E.F
NW2EBC DS OF
* .0.1.2.3.4.5.6.7.8.9.A.B.C.D.E.F
DC X'000102030405060708090A0B0C0D0E0F' 0.
DC X'101112131415161718191A1B1C1D1E1F' 1.
DC X'405A7F7B5B6C507D4D5D5C4E6B604B61' 2.
DC X'F0F1F2F3F4F5F6F7F8F97A5E4C7E6E6F' 3.
DC X'7CC1C2C3C4C5C6C7C8C9D1D2D3D4D5D6' 4.
DC X'D7D8D9E2E3E4E5E6E7E8E9ADE0BD5F6D' 5.
DC X'79818283848586878889919293949596' 6.
DC X'979899A2A3A4A5A6A7A8A9C06AD0A100' 7.
DC X'00000000000000000000000000000000' 8.
DC X'00000000000000000000000000000000' 9.
DC X'00000000000000000000000000000000' A.
DC X'00000000000000000000000000000000' B.
DC X'00000000000000000000000000000000' C.
DC X'00000000000000000000000000000000' D.
DC X'00000000000000000000000000000000' E.
DC X'000000000000000000000000000000FF' F.
* .0.1.2.3.4.5.6.7.8.9.A.B.C.D.E.F
END

```

---



# 13

## Glossary of Installation-Related Terms

---

### **Adalink**

The teleprocessing-monitor-dependent interface module that connects the application/user to Adabas. The actual module name depends on the environment being used; for example, the module name for linking to a batch or TSO program is ADALNK, and for CICS, the module name is ADALNC. The term “Adalink” refers to the module appropriate for the given environment.

### **address converter**

Adabas stores each database record in a Data Storage block having a relative Adabas block number (RABN). This RABN location is kept in a table called the address converter. The address converters, one for each database file, are stored in the Associator. Address converter entries are in ISN order (that is, the first entry tells the RABN location of data for ISN 1, the 15th entry holds the RABN location of data for ISN 15, and so on).

### **address space**

The storage area assigned to a program task/work unit.

### **BUB**

The block of unreadable blocks. It is contained in the primary ASSO RABN 2 and the mirror ASSO RABN 9 for the primary ASSO, DATA, and WORK; in the primary ASSO RABN 9 and mirror ASSO RABN 2 for the mirror ASSO, DATA, and WORK; and the primary and the mirror PLOGn RABN 1 for PLOGn.

### **communicator**

A routine for communicating between operating systems, making remote targets accessible. Entire Net-work is a communicator.

**database administrator**

Controls and manages the database resources. Tasks include defining database distribution structure and resources, creating and maintaining programming and operation standards, ensuring high performance, resolving user problems, user training, controlling database access and security, and planning for growth and the integration of new database resource applications and system upgrades. Also known as the database analyst.

**ID**

An abbreviation of “target ID”, a unique identifier used for directing Adabas calls to their targets.

**ID table**

A reference data list maintained for all active targets within the boundaries of one operating system. The ID table is located in commonly addressable storage.

**IIBS**

The isolated ID bit string, a 256-bit (32-byte) string contained in the ID table header. Each bit corresponds in ascending order to a logical ID. If the bit has the value 1, the corresponding ID is isolated.

**isolated ID**

The ID of an isolated target, which can be specified by the user as a logical ID. An isolated ID must be greater than zero and less than 256. The isolated ID is interpreted as a physical ID for addressing the target.

**isolated target**

A target called directly by a user.

**logical ID**

A user’s identifier of target(s) to which a message is directed. It must be greater than 0 and less than 256 (either explicitly or implicitly, the content of the first byte of ACBFNR is a logical ID).

**MIRTAB**

The mirror table, which indicates the status of primary RABNs. It is contained in the primary and the mirror ASSO RABN 7 for ASSO, DATA, and WORK; and the primary and the mirror PLOG<sub>n</sub> RABN 1 for PLOG<sub>n</sub>.

**non-DB target**

A target that is not an Adabas nucleus. Access and X-COM are non-DB targets.

**physical ID**

The identifier of a target. It must be greater than 0 and less than 65,536. A database ID (DBID) is a physical ID.

**pseudo-cylinder**

The logical cylinder on an fixed-block-addressed (FBA) device that has no actual DASD cylinder.

**reset**

A flag bit is said to be reset when it contains 0.

**router**

A central routine for communication within the boundaries of one operating system. The routine is called by users with Adalink routines, and by targets with ADAMPM. The router's main purpose is to transfer information between the Adalink and Adabas. The router also maintains the ID table. The BS2000 router is the ADARER module, which is loaded into common memory defined by the ADARUN/ADALNK parameter IDTNAME.

**service**

A processor of Adabas calls and issuer of replies. An Adabas nucleus is an example of a service (see also target).

**set**

A flag bit is said to be set when it contains 1.

**subtask**

A task that is spawned from a parent task.

**target**

A receiver of Adabas calls. A target maintains a command queue, and communicates with routers using ADAMPM. A target is also classified as a service (see definition). The Adabas nucleus is a target.

**user**

A batch or online application program that generates Adabas calls and uses an Adalink for communication.



## Index

---

