Syncsort Optimize IDMS

(EZ-IDMS)

Syncsort IDMS Image User Guide

Release 5.4

Version 1.1

May 2002

Syncsort IDMS Image User Guide
(formerly EZ-Image User Guide)

NOTICE

Syncsort IDMS Image User Guide contains proprietary and confidential material, and is only for use by licensees of the Syncsort Optimize IDMS (EZ-IDMS) proprietary software system. This publication may not be reproduced in whole or in part, in any form, without written permission from Syncsort Incorporated.

IBM and z/OS are trademarks of International Business Machines Corporation. CA, CA-IDMS and IDMS are trademarks of Broadcom. The names of other products or brands in this document are trademarks of their respective holders.

Syncsort Optimize IDMS, formerly called EZ-IDMS is designed to be used by licensed users of Broadcom IDMS for z/OS, as a supplement to this product.

Copyright © 1991, 2020 Syncsort Incorporated. All rights reserved.

PREFACE:

EZ-Image is a component of Cogito's **EZ-DB** package which allows the user to operate as if they have multiple copies of CA-IDMS databases, where in reality there is only one underlying physical database.

The **EZ-Image** component ensures that all updates to the imaged copies of the databases as are written to a **EZ-Image VSAM file**.

EZ-Image operates under CA-IDMS release 10.1 and higher, in the IBM MVS and compatible operating environment.

RELATED DOCUMENTATION:

EZ-DB Overview
EZ-DB Integrated **I**nstallation Guide.
EZ-DB Information and Error Messages.
DB-Allocate User Guide.

Table of Contents

Figures

Summary of Enhancements

The following enhancements were made in release 5.3.

- **VSAM SHRPOOL number for native VSAM.**

  This parameter is used to set the SHRPOOL number used when the VSAM Buffer Pool is built. This would be done to avoid conflicts with CA-IDMS when the DMCL contains a native VSAM area. Default value is 0.

- **DBIMEXTR DATA=ONLY option**

  Optional parameter forces EZ-Extract to extract the data portion of records only without the record prefix.

- **DBIMUTIL Expand Page function.**

  This new function allows an existing EZ-Image database to be resized in the event that the corresponding underlying database has been XPAG'ed.

The following enhancements were made in release 5.2.

- **EZ-Extract Utility**

  EZ-Extract provides the ability to perform simple regression tests by allowing users to compare the contents of image files after individual tests. For example, current procedure may be to run a test, backup the database, run converted application, compare databases using tools such as FILE AID,

  Image Extract provides the ability to perform simple regression tests by allowing users to compare the contents of two image files at the logical level. For example,

  - A user runs tests using EZ-Image with no aging
  - Run EZ-Extract utility to deblock all EZ-Image data into sequential file, this will call database procedures, decompression routines etc, so you end up with a sequential file containing one entry per database record prefixed by dbkey, record id.
  - Init EZ-Image VSAM file
  - Repeat tests with Aging
  - Extract aged data
  - Run file compares of the two sets of aged data.

The following enhancements were introduced in release 4.3.

- **Improved VSAM file utilisation statistics**

- **Specify image file and CA-IDMS file open mode**

The following enhancements were introduced in release 4.2.

- **Hot Backup Option (EZ-HotBackup)**

  Hot Backup is the ability to take a back-up of a database area while that database area is fully available to end-users for update. This is a highly desirable feature for users wishing to provide 24-hour or extended availability. Although previously available using standard CA-IDMS facilities, most users do not take advantage of this functionality because it is perceived to be complex and un-reliable.

Now, with EZ-Image ***EZ-HotBackup*** option, Hot Backups become so simple that they will become standard procedure for all EZ-Image users.

- **Duplex Database Option** (**EZ-Duplex**)

  Database duplexing is the implementation of a mirror database to be used in the event of a failure of the production database. Its purpose is to increase system resilience and reduce down-time in the event of a failure (of the prime database). In the past, this was accomplished by taking a full copy of the database and mirroring updates using the IDMSDPLX exit. Very few users took advantage of this function because of the cost of the required DASD and the need for system exits.

  EZ-Image ***EZ-Duplex*** option provides a simple and more cost effective Duplex functionality without any need for user exits and a drastically reduced DASD requirement.

- **Dynamic Allocation Enhancements**

  It is no longer necessary to include DD statements for the VSAM image files in the CV startup or local mode JCL.

Introduction

This manual describes the operation of the **EZ-Image** product from **Cogito Limited**. It should be read by the CA-IDMS Database Administrator and by the installer of **EZ-Image**.

Section 1 provides an overview of **EZ-Image**.

Section 2 provides details of the **EZ-Image** environment.

Section 3 describes the **EZ-Image** operation.

Section 4 provides details on how to install **EZ-Image**.

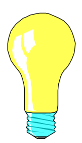
Section 5 provides details on the run-time JCL requirements.

Section 6 provides details on how to implement and use the **EZ-HotBackup** and **EZ-Duplex options.**

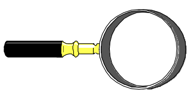
Section 7 provides details about the **EZ-Extract** utility.

Section 8 provides various technical notes tips on tuning and other issues.

Several symbols are used to draw attention to various topics. The topics and their use is as follows:



Light Bulb: An idea. For instance, an example on how a particular feature might be used in practise.



Magnifying Glass: A more detailed technical explanation of a particular subject.

Stop Sign: Ignore this information at your peril!

Features and Benefits Overview

1 Features and Benefits Overview

EZ-Image provides the ability to operate as if you have multiple copies of a CA-IDMS database, while in reality there is only one copy of the underlying physical database.

With **EZ-Image,** you will no longer need to create separate test databases for individual users. Instead, each user (or group of users) is allocated a **EZ-Image VSAM file** to which that users updates are stored.

In this way, each user operates as if they have exclusive use of the entire underlying CA-IDMS database without the cost of actually creating multiple databases.

You may run **EZ-Image** against test or production underlying databases.

EZ-Image will immediately bring your site the following benefits:

Test against Production	Now your test runs can execute against the full production database - instead of a cut-down subset. This is ideal for replication of problems which may not occur in the test environment. Test programs can execute in update mode against the production databases without changing the data.
Realistic End-User Training	Instead of setting up training databases for end-users, they may now train against the real production database (with update capability) without compromising the integrity of the real data.
Save DASD Space	Physical DASD space is an expensive resource. How much space does your site have tied up providing a multitude of different test environments. **EZ-Image** saves space because you only need one copy of the underlying physical database. Each EZ-Image VSAM file contains only the modified pages (in compressed format).
No need for database Restores	For repetitive testing or to restart a failed job - just use the online commands or the batch utility program to initialise the applicable **EZ-Image** VSAM file! No need to wait for a database restore.

Save Valuable DBA Time

How much time do your DBA's spend maintaining multiple test databases - including definition, population and recovery?

With **EZ-Image** these chores disappear - freeing your DBA's valuable time.

New **EZ-Image** databases are dynamically created and implemented. There is no need to make schema or DMCL changes.

You should be aware that, as only updates are written to the EZ-Image files, the underlying physical database should remain static. That is, if updates are also performed to the underlying physical CA-IDMS database, then unpredictable results could be produced on the virtual databases due to broken chains or corrupt indexes etc.

EZ-HotBackup Option

Use the ***EZ-HotBackup*** option to take backups of your production databases while they are fully available for retrieval **and update**.

EZ-Duplex Option

Use the ***EZ-Duplex*** option to provide simple and cost effective mirror databases.

EZ-Extract Option

Use the ***EZ-Extract*** utility to perform simple comparisons of the results of test runs by comparing only the modified CA-IDMS records extracted from the applicable VSAM image files.

EZ-Image Environment

2 EZ-Image Environment

2.1 DBA Overview

EZ-Image implements *imaged* CA-IDMS databases through the utilization of alternate page-groups. If you are not familiar with the concept of alternate page-groups, you should refer to the applicable CA-IDMS documentation.

If a user wishes to operate against an imaged version of a particular database, the DBA will determine the page-group of the CA-IDMS file to be imaged (referred to as the base page group) and assign that user (or group of users) a new page-group number referred to as the image page group.

EZ-Image provides a utility to automatically generate the new subschema modules required, from the base subschema load modules. In addition, at run-time, **EZ-Image** will dynamically expand the DMCL to include the definitions required to access the imaged CA-IDMS files.

Furthermore, there is no need to include DD statements in the run-time DMCL for the imaged database files. These are dynamically allocated as required by the DB-Allocate component of EZ-DB.

The steps required to be performed by the DBA to implement a **EZ-Image** database are as follows:-

- Install the base **DB-90s** software, and generate a COXATAB specifying IMAGE=YES.
- Generate a **COIMAGE** table to define the EZ-Image database(s). Note. As a alternative to this step, the EZ-Image database(s) may be defined at run-time through an in-stream parameter file (DBIMCTL).
- Run an IDCAMS job to allocate and initialize the EZ-Image VSAM file(s).
- Run the COIMZAPS utility to generate new subschema modules as required.
- Modify the run-time JCL to include required EZ-Image parameters.
- Generate a new DBNAME table to allow access to the new imaged databases.

These steps are described in detail in Section 4 - Installation and Generation.

2.2 Operation Overview

This gives an overview of how **EZ-Image** operates and works.

After EZ-Image has been installed, at CV startup, or local mode job start, EZ-Image's initialisation performs the following:

When the DMCL is loaded, EZ-Image reads the COIMAGE table looking for an appropriate DMCL entry -

alternatively it can read the COIMCTL instream parameters.

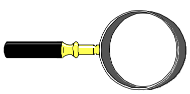
It then dynamically expands the DMCL and builds the appropriate area and file control blocks for the image databases you have requested.

Once the DMCL has been expanded, CA-IDMS sees the imaged files and areas as being normal CA-IDMS files and areas.

Any normal CA-IDMS function can be carried out against these imaged files and areas.

For example the full range of DCMT commands can be used against them.

2.2.1 EZ-Image Read Processing

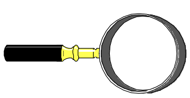


When CA-IDMS requests a page to be read from an imaged file, the EZ-Image software intercepts the IO and checks whether the page required has been written to the EZ-Image VSAM file. If it has, the page is read from the VSAM file, decompressed and moved into the CA-IDMS buffer.

If the page is not on the VSAM file, the read IO is converted to read the page from the underlying physical database file.

Either way, CA-IDMS believes it has read a standard IDMS page from the image database file.

2.2.2 EZ-Image Write Processing



When CA-IDMS requests a page to be written to disk from an imaged file, the EZ-Image software intercepts the IO. The IDMS page is compressed before being written to the EZ-Image VSAM file.

CA-IDMS believes it has written a standard page to the virtual database file.

2.3 CA-IDMS Journals

This section only applies if you have installed EZ-Image on a CA-IDMS Release 10.2 system.

When EZ-Image is first installed, or whenever you add new imaged databases, your CA-IDMS journal files should always be initialised.

This is due to a standard CA-IDMS requirement that area information that is stored in the journal header records (JHDAs). CA-IDMS Release 10.2 is not capable of successfully expanding these JHDA records. When the expanded DMCL is built, there is no room for the new areas in the existing JHDAs.

By initialising the journals, space is made available, so that the first time CV is started with EZ-Image active, the JHDA records can be expanded.

NOTE. This only applies to systems running on CA-IDMS Release 10.2. Release 12.0 systems do not need to have their journals initialised, as CA-IDMS is able to successfully expand the journal headers.

If you do not initialise your CA-IDMS release 10.2 journals, you may get warmstart and recovery failures on the new imaged databases.

EZ-Image Operation

3 EZ-Image Operation

3.1 Using the EZ-Image Online Display Program

EZ-Image can be monitored online using the supplied program COIMDA01. Using the default task code DBIM to display the current status of your CV's EZ-Image files and page groups. On 3270 type terminals, the display will be in full colour. If your terminal is not a 3270, but does support extended colour attributes, you can use the task code DBIMC, which will force full colour displays.

NOTE: If you use DBIMC on terminals which do not support extended colour attributes, unpredictable displays may be produced!

3.2 EZ-Image File Summary Display

The EZ-Image File Summary screen is the first screen displayed after invoking the DC/UCF task code DBIM or DBIMC. The display allows you to monitor the current statistics and settings of all the EZ-Image IDMS files created.

```
COGITO EZ-IMAGE R4.1                                    V0005  16:55:50 94.354
                *** EZ-IMAGE FILE SUMMARY DISPLAY ***
EZ-IMAGE Table : COIMAGE    # Data Buffs     100  # IMAGE VSAM Files :     4
DMCL Name      : DMGLBD2    # Indx Buffs     150  # IMAGE IDMS Files :    26
                                                  # IMAGE PGs        :     3
   -----Image----  -----IDMS-----  VSAM               ---VSAM Totals---
   File        PG  File        PG  DDNAME    Sta Comp   #Reads  #Writes
 ----------------------------------------------------------------------------
   DF00166     66  DF00152     52  COIM6601  ACT FREE      123       23
   DF00266     66  DF00252     52  COIM6601  ACT FREE     1078      116
   DF00366     66  DF00352     52  COIM6601  ACT FREE     1123      109
   DF00466     66  DF00452     52  COIM6601  ACT FREE      789       76
   DF00566     66  DF00552     52  COIM6601  ACT FREE     1567      144
   DF00666     66  DF00652     52  COIM6601  ACT FREE     1345      111
   DICTDB01     1  DICTDB       0  COIMPG1   ACT FREE       23        1
   DLODDB01     1  DLODDB       0  COIMPG1   ACT FREE        1        1
   DLOGDB01     1  DLOGDB       0  COIMPG1   ACT FREE        0        0
 ----------------------------------------------------------------------------
   Totals                                                 9563      765

<CLEAR> EXIT   <PF7> UP   <PF8> DOWN   <PF12> MODE
ENTER COMMAND BELOW>
```

Figure 1 EZ-Image File Summary Screen

Figure 1 shows the online display in response to the DBIM task. The information shown is detailed below;

Header Section: The display has a header section containing the following information:

EZ-Image Table: The name of the COIMAGE table currently being used by this CV.

DMCL Name: The name of the global DMCL being used by the CV. The DMCL name will determine which entries in the COIMAGE table are used for this CV.

 # Data Buffs: This shows the number of VSAM buffers currently allocated for EZ-Image to use for the data portion of the Image VSAM files. This value is defined in the DMCL entry of the COIMAGE table.

 # Indx Buffs: This shows the number of VSAM buffers currently allocated for EZ-Image to use for the Index portion of the Image VSAM files. This value is defined in the DMCL entry of the COIMAGE table.

 # VSAM Files: This shows the total number of EZ-Image VSAM files being used by this CV.

 # IDMS Files: This shows the total number of EZ-Image IDMS files that have been created for this CV. A EZ-Image IDMS file will be created for each real IDMS file that is to be imaged for each EZ-Image page group.

 # IMAGE PGs: This is the number of EZ-Image page groups that have been created for this CV.

Summary Section: For each EZ-Image file that has been created, an entry is displayed in this section.

Image File: This is the name of the file that EZ-Image has created, having applied the File Key and Mask values to the CA-IDMS file that is being imaged.

Image PG: This is the EZ-Image Page Group that the EZ-Image file belongs to.

IDMS File: This is the name of the real CA-IDMS file that is being imaged.

IDMS PG: This is the CA-IDMS Page Group being imaged.

VSAM DDNAME: This is the DDNAME of the EZ-Image VSAM file being used to store updates to this EZ-Image file. If multiple DDNAMEs were specified for the EZ-Image page group in the COIMAGE table, only the first DDNAME will be displayed here.

Sta: This shows the current status of the EZ-Image file. Values are 'ACT' - showing the file is Active and in use, 'OFF' - showing the file has been varied offline from EZ-Image. Any attempted access to the EZ-Image file by IDMS, will be perceived as an IO error.

Comp: This shows the type of compression being used by EZ-Image on this file, when pages are written to the EZ-Image VSAM file. the values are FREE - indicating that free space within the IDMS page will be compressed, MAX - indicating that all repeated full words within the IDMS page will be compressed.

VSAM # Reads: This shows the total number of read IOs that were performed against the EZ-Image VSAM files, for this EZ-Image file, since CV startup.

VSAM # Writes: This shows the total number of write IOs that were performed against the EZ-Image VSAM files, for this EZ-Image file, since CV startup.

Totals Section: Totals are shown for the total number of VSAM read and write IOs. These figures are an accumulation for all EZ-Image files defined to the CV, not just those shown on the current page of the display.

PF-Key Usage: The following defines the actions assigned to the various PF-Keys that may be used within the DBIM task:

<ENTER> The <ENTER> key refreshes the display with current information.

<CLEAR> The <CLEAR> key is used to terminate the display and return to the DC/UCF ENTER NEXT TASK CODE prompt.

<PF7> The <PF7> key is used to page UP through the display (if there are more EZ-Image files than can be displayed on one screen).

<PF8> The <PF8> key is used to page DOWN through the display (if there are more EZ-Image files than can be displayed on one screen).

<PF12> The <PF12> key is the default MODE key that is used to switch between the different EZ-Image displays. Note that this key can be modified within the COXATAB table, if you wish to use a PF key other than PF12.

3.3 EZ-Image Page Group Display

The EZ-Image Page Group screen is displayed by pressing the MODE key (default is PF12), from the File Summary display. The display allows you to monitor the current status and statistics of all the EZ-Image page groups that have been setup for this CV.

Figure 2 shows the online EZ-Image Page Group Summary screen. The information is detailed below:

```
COGITO EZ-IMAGE R4.2                                           V0005  16:55:58 94.354
                *** EZ-IMAGE PAGE GROUP DISPLAY  ***
EZ-IMAGE Table : COIMAGE    # Data Buffs     100  # IMAGE VSAM Files :     4
DMCL Name      : DMGLBD2    # Indx Buffs     150  # IMAGE IDMS Files :    26
                                                  # IMAGE PGs        :     2
   -----Image----  -----IDMS-----           #   1st VSAM  ---VSAM Totals---
      PG  Segment    PG   Segment  Comp   DD   DDNAME      #Reads  #Writes
 -----------------------------------------------------------------------------
       1 IMAGDICT     0  APPLDICT  FREE    1   COIMPG1         24        2
      66 IMAGPG66    52     DBD52  FREE    2   COIM6601      9539      763









------------------------------------------------------------------------------
   Totals                                                    9563      765

<CLEAR> EXIT   <PF7> UP   <PF8> DOWN   <PF12> MODE
ENTER COMMAND BELOW>
```

Figure 2 EZ-Image Page Group Summary Screen

Header Section: The display has a header section containing the same information as the File Summary Display Header Section.

Summary Section: For each EZ-Image Page Group that has been created, an entry is shown on this display.

Image Segment: This column is only shown if the CV is running under CA-IDMS Release 12.0. It shows the name of the EZ-Image Segment as defined in the Page Group entry of the COIMAGE table.

Image PG: The EZ-Image page group that images the IDMS page group, as defined in the COIMAGE table.

IDMS Segment: This column is only shown if the CV is running under CA-IDMS Release 12.0. It shows the CA-IDMS segment that is being imaged as defined in the Page Group entry of the COIMAGE table.

IDMS PG: The CA-IDMS page group that is being imaged, as defined in the COIMAGE table.

Comp: This shows the type of compression being used on the records being written to the Image VSAM file. FREE indicates that the FREE space on the IDMS page will be compressed, MAX indicates that all repeating fullwords on the IDMS page will be compressed.

 # DD: Shows how many EZ-Image VSAM files have been defined to this EZ-Image page group.

1st VSAM DDNAME: This shows the first EZ-Image VSAM file DDNAME that is used by this EZ-Image Page Group.

VSAM # Reads: This shows the total number of read IOs that were performed for this EZ-Image page group.

VSAM # Writes: This shows the total number of write IOs that were performed for this EZ-Image page group.

Totals Section: Accumulated totals of all EZ-Image VSAM read and writes IOs are shown.

PF-Key Usage: The actions assigned to the various PF-Keys are described under the EZ-Image File Summary Screen section.

3.4 EZ-Image VSAM File Display

The EZ-Image VSAM File screen is displayed by pressing the MODE key (default is PF12), from the Page Group Summary display. The display allows you to monitor the current status and statistics of all the EZ-Image VSAM files that are defined to this CV.

```
COGITO DB-IMAGE (R4.3)                                     V0070  10:18:35 97.258
                  ***  DB-IMAGE VSAM FILE DISPLAY  ***
  DB-IMAGE Table : COIMAGE    # Data Buffs     100  # IMAGE VSAM Files :     47
  DMCL Name    :    DMGLBDV    # Indx Buffs     150  # IMAGE IDMS Files :    235
                                                    # IMAGE PGs :           235
   VSAM          #IMAGE    ------------Data------------ --------Index--------
   DDNAME   Sta   Files    Records Xtnts  K-Used   K-Avl Xtnts  K-Used   K-Avl
 -------------------------------------------------------------------------------
 R132927    OPN       6        339    2     2760   39450     3      26      52
 R132917    OPN       4        331    1     2760   32520     2      26      47
 R132027    OPN       5        598    6    12420   62040     7      33      65
 R132017    OPN       5        128    4      690   55170     5      25      64
 R131717    OPN       5        128    1      690   34470     2      25      49
 R130717    OPN       5        128    1      690   34470     2      25      49
 R130427    OPN       5        128    1      690   34470     2      25      49
 R130417    OPN       5        128    1      690   34470     2      25      49
 R130227    OPN       5        128    4      690   55170     5      25      64
 -------------------------------------------------------------------------------


 <CLEAR> EXIT   <PF7> UP   <PF8> DOWN   <PF09> MODE
 ENTER COMMAND BELOW>
```

Figure 3 EZ-Image VSAM File Display

Figure 3 shows the online EZ-Image VSAM File screen. The information is detailed below:

Header Section: The display has a header section containing the same information as the File Summary Display Header Section.

Summary Section: For each EZ-Image VSAM File that is in use by this CV, an entry is shown on this display.

VSAM DDNAME: Shows the DDNAME of each EZ-IMAGE VSAM file.

Sta: Displays the current status of the EZ-Image VSAM file.

#IMAGE Files: Indicates how many EZ-Image Files are mapped to this VSAM file.

The display then details the VSAM file utilization for both the data areas and the associated VSAM indexes.

------------ Data ------------

Records Number of data records (i.e., CA-IDMS pages) written to this VSAM file.

Xtnts Number of extents currently allocated.

K-Used Ending RBA of space used in data component in bytes.

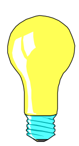
K-Avl Amount of available space in data component in bytes.

------------ Index ------------

Xtnts Number of extents currently allocated.

K-Used Ending RBA of space used in index component in bytes.

K-Avl Amount of available space in index component in bytes.



VSAM will automatically allocate additional extents as required up to a maximum of 123. However, you should monitor the usage and available disk space and consider reallocating the file when appropriate.

PF-Key Usage: The actions assigned to the various PF-Keys are described under the EZ-Image File Summary Screen section.

3.5 Online Commands

Each on-line display screen has a command line at the bottom of the screen. You can use the command line to perform a number of functions against the EZ-Image files and page groups.

Commands are available to perform the following functions:-

1. Vary EZ-Image Page Groups offline and online.

2. Close a EZ-Image VSAM file.

3. Initialise a EZ-Image Page Group's VSAM files.

4. New Copy EZ-Image Segments.

All of the commands described below can also be invoked directly from the DC/UCF *enter next task code* prompt using the **DBIMV** task code. This allows you to enter command streams using the UCFBATCH facility if desired. To use the DBIMV task, simply enter the task code DBIMV followed by the syntax described below. For example,

```
DBIMV VARY IMAGEPG=99 OFFLINE
```

The syntax for each available function is as follows. Note that keywords are separated by spaces;

3.5.1 Vary Image Database Offline/Online

```
VARY IMAGEPG=99 OFFLINE
```

where ***99*** is the Page Group that you wish to vary offline.

This command will vary each EZ-Image File associated with the EZ-Image Page Group you specify offline. If any access is subsequently attempted to a file that is offline, an I/O error will be returned to CA-IDMS.

```
VARY IMAGEPG=99 ONLINE
```

where ***99*** is the Page Group that you wish to vary back online.

This command will reverse the effect of the 'VARY OFFLINE' and make each EZ-Image file associated with the page group active again.

3.5.2 Close an Image VSAM File

```
VARY VSAM=vsfile CLOSE
```

where '***vsfile***' is the name of the EZ-Image VSAM file you wish to close.

This command closes the VSAM file, but does not make it offline. The next time an IO is performed against the VSAM file, it will be automatically reopened. The main use of this command is when you wish to initialise a VSAM file while the CV is still active - see below.

3.5.3 Intialise an Image Database

```
VARY IMAGEPG=99 INIT
```

where ***99*** is the EZ-Image Page Group you wish to initialise.

This command will initialise each of the VSAM files that have been assigned to this EZ-Image Page Group, provided the VSAM file has first been closed. If the VSAM file is still open, then the command will fail. Once the file has been initialised there is no way to restore the data that was previously held on it, so ensure you really do want it initialised!

If you have VSAM files which have multiple EZ-Image Page Groups assigned to them. Be careful with the use of the INIT Page Group online command, as it will initialise the entire VSAM file, not just the one page group specified.

3.5.4 New Copy EZ-Image Segments

```
NEWCOPY
```

When a DCMT VARY DMCL NEWCOPY is issued under CA-IDMS release 12.0, it is necessary to issues the EZ-Image NEWCOPY command to re-build the EZ-Image segments based upon the modified DMCL structure. This should be done immediately after the DCMT VARY DMCL NEWCOPY has successfully completed.

3.6 DBIMUTIL (Offline Utility)

EZ-Image is supplied with a utility program - **DBIMUTIL** - that performs a number of functions against both the EZ-Image VSAM files and the CA-IDMS database files.

DBIMUTIL runs as a stand alone batch program. Note that it is **not** an IDMS local mode program. Details of the JCL required to run **DBIMUTIL** are given later in this section.

The functions that **DBIMUTIL** performs are;

1. **UNLOCK** a EZ-Image area.

2. **INITIALISE** an entire EZ-Image VSAM file.

3. **INITIALISE** a particular EZ-Image Page Group, from a specified EZ-Image VSAM file.

4. **COPY** one EZ-Image Page Group from a EZ-Image VSAM file onto another EZ-Image VSAM file, and change the page group.

5. **UPDATE** an IDMS page group with the images from a EZ-Image page group.

6. **EXPAND** the page size of a specified EZ-Image page group.

Each of these functions will be described in further detail along with the syntax of the command line required to perform it.

Note that all parameters are separated by at least one space.

3.6.1 Unlock

If EZ-Image is being used in a local mode job, which abends, then any imaged areas will be locked, in the same way that the CA-IDMS areas are locked. Because the page containing the area lock will have been written to the EZ-Image file, we cannot use IDMSPFIX (or IDMSBCF for Release 12.0) to unlock the area. Instead DBIMUTIL must be used.

DBIMUTIL requires the following parameters to be able to unlock the area;

IMAGEPG: The EZ-Image page group containing the area that you wish to unlock.

AREA: The full area name as defined in the CA-IDMS DMCL of the base IDMS area. For Release 12.0 users, this includes the IDMS segment name prefix.

DMCL: The name of the CA-IDMS DMCL module that contains details of the base IDMS area specified in the above parameter. Note that DBIMUTIL will need to load the DMCL module, so ensure that the library that contains the DMCL load module is correctly included in the jobs STEPLIB - See JCL requirements for further details.

The full syntax for the **UNLOCK** function is:

UNLOCK IMAGEPG=*nn* AREA=*areaname* DMCL=*dmclname*

Example;

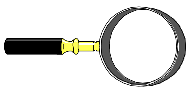
```
UNLOCK IMAGEPG=66 AREA=DBUSER.CUSTOMER-AREA DMCL=R120DMCL
```

Where **66** is the EZ-Image page group that the area to be unlocked is in, and DBUSER.CUSTOMER-AREA is the base IDMS area name - in this example it is qualified with the Release 12.0 segment name.

3.6.2 Initialise VSAM File

You can initialise an entire EZ-Image VSAM file using this option. If the VSAM file is used by more than one EZ-Image Page Group, then ensure that you want the entire file initialised. If you only want a particular page group to be initialised, then use the selective initialise detailed later.

The only parameter required to initialise the entire VSAM file, is the DDNAME you have assigned to the EZ-Image VSAM file for this job.



Refer to the Technical Note section for considerations regarding image pages that may still be present in the CA-IDMS buffers.

The syntax for the VSAM file INITIALISE function is:

INIT DDNAME=*ddname*

Example;

```
INIT DDNAME=VSAM01
```

3.6.3 Initialise Page Group

You can initialise a particular EZ-Image page group on a specific VSAM file using this option.

The only parameter required is the EZ-Image Page Group number that you want to initialise. The EZ-Image file that you wish the page group to be removed from, must be defined as INDD - see JCL requirements for full details.

If the EZ-Image Page Group is contained across multiple VSAM files, you will need to run the Page Group Init for each VSAM file.

The syntax for the Page Group Initialise is:

INIT IMAGEPG=*nn*

Example;

```
INIT IMAGEPG=66
```

3.6.4 Copy Image Page Group

This function allows you to copy a specified EZ-Image Page Group on one VSAM file to a different Page Group, on a different VSAM file.

Two parameters are required;

IMAGEPG	Specifies the current EZ-Image page group that resides on the EZ-Image VSAM file allocated to the DDNAME - INDD.
NEWPG	Specifies the EZ-Image page group that the records are to be copied to on the EZ-Image VSAM file allocated to the DDNAME - OUTDD.

Note - If a particular page already exists on the OUTDD file, then it will be replaced, otherwise a new record will be added to the VSAM file.

This option is particularly useful if you wish to create new EZ-Image page groups based upon work which has already been carried out, and which has used EZ-Image to create a base set of data.

If the EZ-Image Page Group is contained on multiple VSAM files, then you should run the COPY for each VSAM file that contains the Image Page Group.

The full syntax for the COPY function is;

```
COPY IMAGEPG=nn NEWPG=nn
```

Example;

```
COPY IMAGEPG=66 NEWPG=88
```

3.6.5 Update IDMS

This function can be used to transfer records from a EZ-Image VSAM file for a particular EZ-Image page group, onto the real CA-IDMS files, for the real CA-IDMS page group.

This option is useful when work has been carried out on an imaged database which you then wish to use as the base for all future work, as it allows you to update the CA-IDMS database files with your modified data that was stored on the EZ-Image VSAM file.

Several parameters are required for the Update IDMS Function;

IMAGEPG: The EZ-Image page group that you wish to copy from. The VSAM file that the page group is to be copied from must be defined to the ddname INDD.

IDMSPG: The CA-IDMS page group that you wish to update.

DMCL: The CA-IDMS DMCL load module name, that contains the details of the CA-IDMS page group you wish to update. Any CA_IDMS database files that maybe updated by this function must be defined in the JCL - see JCL requirements.

If the EZ-Image Page Group is contained on multiple VSAM files, then you should run the UPDIDMS for each VSAM file that contains the Image Page Group.

The full syntax for the UPDATE IDMS function is:

UPDIDMS IMAGEPG=*nn* IDMSPG=*nn* DMCL=*dmclname*

Example;

```
UPDIDMS IMAGEPG=66 IDMSPG=52 DMCL=R120DMCL
```

3.6.6 Expand Page Size

This function allows you to perform an EXPAG against an EZ-Image database. For example, if you expand the page size of an underlying database against which there are one or more EZ-Image database(s), you should use this function against each applicable image database to keep the page sizes in step.

The following parameters are required for the Expand Page functions:

NEWSIZE: The new page size.

IMAGEPG: The EZ-Image page group that you wish to re-size. The VSAM file relating to this image must be defined to the ddname INDD

FILE: The segment.filename of the base file. Note this must be the name of the underlying base database file, not the EZ-Image file name.

DMCL: The name of the applicable DMCL module. Note: The DMCL loaded must be the original DMCL with the old page size.

This function of DBIMUTIL will increase the free space count on all the pages for the file specified by the *difference* between the file page size in the DMCL, and the new page size specified. Note that the free space within the image file is compressed.

The full syntax of the EXPAND Page function is:

EXPAND NEWSIZE=*nn* IMAGEPG=*nn* FILE=*segment.filename* DMCL=*dmcl-name*

For example,

```
EXPAND NEWSIZE=8192 IMAGEPG=66 FILE=SYSTEM.DCDML DMCL=R120DMCL
```

3.6.7 DBIMUTIL Usage Considerations

There are a number of points that should be considered prior to setting up DBIMUTIL to run.

If you are initialising a page group that is contained on more than one EZ-Image VSAM file, then you should ensure that all the VSAM files are initialised together, otherwise you may encounter problems such as broken chains etc.

Similarly if you are copying one EZ-Image page group to another, that spans multiple VSAM files, again you should ensure each VSAM file is copied in turn.

If you are using the Update IDMS function for a EZ-Image page group that spans multiple VSAM files, ensure that you run the function against each of the VSAM files, to avoid corrupting the IDMS database files.

3.6.8 DBIMUTIL JCL Requirements

The JCL required is straightforward and an example is shown in **Figure 4**.

```
//RUNUTIL  EXEC PGM=DBIMUTIL,REGION=1024K
//STEPLIB  DD DSN=cogito.loadlib,DISP=SHR
//         DD DSN=user.dba.loadlib,DISP=SHR
//INDD     DD DSN=user.image.vsam1,DISP=SHR
//OUTDD    DD DSN=user.image.vsam2,DISP=SHR
//DF00100  DD DSN=caidms.database.file1,DISP=SHR
//DF00200  DD DSN=caidms.database.file2,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
UNLOCK IMAGEPG=01 AREA=CUSTOMER-AREA DMCL=R120DMCL
INIT IMAGEPG=66
COPY IMAGEPG=01 NEWPG=02
UPDIDMS IMAGEPG=01 IDMSPG=00 DMCL=R120DMCL
INIT DDNAME=INDD
/*
```

Figure 4 DBIMUTIL JCL

The information shown in bold lower case should be modified as follows:

cogito.loadlib: The name of the library into which you unloaded the EZ-DB installation tape load library. You must ensure you are using the correct library for your version of CA-IDMS.

user.dba.loadlib: The name of the load library that contains the DMCL load module. Note that this is only required if you are using the UNLOCK or UPDIDMS functions.

user.image.vsam1: The name of the EZ-Image VSAM file to be used as input to ddname INDD. Its use depends on the function being run:

UNLOCK INDD must be the VSAM file that contains the first page in the area.

INIT INDD is the VSAM file to be initialised. For the full VSAM file initialisation option, a ddname other than INDD can be used, and specified on the INDD=*ddname* parameter.

COPY INDD is the VSAM file from which any pages that match the specified page group are copied FROM.

UPDIDMS INDD is the VSAM file from which any pages that match the specified page group are copied to their required CA-IDMS database file.

user.image.vsam2 The name of the EZ-Image VSAM file to be updated must be assigned to the ddname OUTDD. This DD card is only required for the following functions:

COPY OUTDD is the VSAM file to which the new page group records will be copied.

In addition to the above changes, if you are using the UPDIDMS function, you must include DD statements for each CA-IDMS database file that may be required for updating. In the above example DF00100 and DF00200 are supplied as they will be updated by the UPDIDMS statement.

You should ensure any required CA-IDMS database file is allocated to the ddname as specified in the DMCL being used.

Confirmation messages, or error messages are written to the SYSPRINT file. Upon completion, check the SYSPRINT output for successful completion, or for any problems that were encountered during the job.

3.7 Shutdown Reports

When using **EZ-Image**, a report is written to the **DBIMLOG** file, at job termination or CV shutdown.

A sample portion of a typical report is shown in **Figure 5.**

```
   7 DEC 94  9:38               EZ-IMAGE VSAM FILE STATISTICS                         PAGE 001
                         JOB: CHIDCP1M DMCL: R120DMCL COIMCTL: COIMAGE
         DATA BUFFERS      100 CISIZE 18432   INDEX BUFFERS      150 CISIZE   512

FILE          IDMS EZ-IMAGE HIGH PAGE COMPRESS IGNORE  READ I/O  UPDATE I/O IDMS PAGES
NAME       SEGMENT  SEGMENT                    PGLOCK                          ON FILE

COIMPG1  APPLDICT IMAGDICT          0 FREE     YES          124         104         28
COIMPG2  DBD52    DBD66             0 FREE     YES         1897         456        311
```

Figure 5 Job/Step Shutdown EZ-Image Report

3.7.1 Report Header

Line-1: The first line of each page commences with the date/time of the report and the page number.

Line-2: This line shows the job or CV name, the DMCL being used and the COIMAGE table being used.

Line-3: This line contains the following information:

DATA BUFFERS
This is the number of VSAM buffers allocated for the data portion of the VSAM file, as defined in the DMCL entry of the COIMAGE table.

CICSIZE
This is the VSAM control interval size of the data portion of the VSAM file.

INDEX BUFFERS
This is the number of VSAM buffers allocated for the index portion of the VSAM file, as defined in the DMCL entry of the COIMAGE table.

CISIZE
This is the VSAM control interval size of the index portion of the VSAM file.

3.7.2 Detail Lines

For each EZ-Image VSAM file that was allocated to this job or CV, a detail line is written giving the following information:

FILE NAME
The EZ-Image VSAM file's ddname.

IDMS SEGMENT
For CA-IDMS Release 12.0 only, the CA-IDMS segment that was being imaged to this VSAM file is shown - For Release 10.2 users this is replaced by the IDMS Page Group number that is being imaged to this VSAM file.

EZ-IMAGE SEGMENT
For CA-IDMS Release 12.0 only, the EZ-Image segment name that has been created, and which is imaged to this VSAM file is shown - For Release 10.2 users, the EZ-Image Page Group number that has been created, is shown instead.

HIGH PAGE
If you have split a EZ-Image database file across multiple VSAM files, then the high page number that is imaged to this VSAM file is shown, otherwise this column is left blank.

COMPRESS
This shows the type of compression being used on the IDMS pages being written to the VSAM file. FREE indicates that any free space on the IDMS page was compressed, MAX indicates that all repeating full words on the IDMS page were compressed.

IGNORE PGLOCK

Shows whether the page locks encountered on the real IDMS database files were ignored or not.

READ I/O
Shows the total number of read I/Os performed against this VSAM file.

WRITE I/O
Shows the total number of write I/Os performed against this VSAM file.

IDMS PAGES ON FILE
Shows the current number of imaged IDMS database pages that are stored on the VSAM file.

EZ-Image Installation & Generation

4 EZ-Image Installation & Generation

4.1 The EZ-DB COXATAB Table

EZ-Image is supplied as part of the Cogito EZ-DB integrated installation tape, and it is assumed that you have already completed the base EZ-DB installation. Refer to the EZ-DB Integrated Installation guide for full details.

The EZ-DB installed SRCLIB contains a member **INSTIMAG**, that contains skeleton JCL for performing the various tasks defined in this section.

When installing EZ-DB, you will have generated a COXATAB table specifying which components of EZ-DB are to be activated.

To activate the EZ-Image component of EZ-DB, you must code the IMAGE=YES (and optionally the HOTBACK=YES parameter) when generating your COXATAB table as illustrated in the following figure (**Figure 6**).

```
       #COXA IMAGE=YES
             HOTBACK=YES
```

Figure 6 #COXA macro parameters.

4.2 Generating the COIMAGE table

The COIMAGE table is generated by the DBA to define the relationships between the BASE (underlying) CA-IDMS databases and the EZ-Image databases.

You may define all of the sites EZ-Image databases in one COIMAGE table. Alternatively, you may generate COIMAGE tables for particular jobs or even specify the parameters through a run-time control file. Refer to section 5.2 for details of using the DBIMCTL run-time parameters.

The default name of the COIMAGE table is 'COIMAGE'. You may however use an alternative name which should be specified through the COIMAGE= parameter in the COXACTL control file.

Refer to the EZ-DB Integrated Installation guide for details of the COXACTL file.

The COIMAGE table is generated using the #COIMAGE macro. The following illustrates the three classes of #COIMAGE macro which may be combined to generate a COIMAGE table:

```
#COIMAGE    ENTRY=DMCL,<dmcl-options>
#COIMAGE    ENTRY=PG,<page-group-options>
#COIMAGE    ENTRY=LAST
```

To generate a valid COIMAGE table, you would code one or more #COIMAGE ENTRY=DMCL statements, each followed by one or more ENTRY=PG statements. The ENTRY=PG statements relate to the DMCL defined by the prior ENTRY=DMCL statement. The table is always terminated by an ENTRY=LAST statement.

At run-time, EZ-IMAGE will search the COIMAGE table for an ENTRY=DMCL statement relating to the run-time DMCL and activate the applicable EZ-Image files for the defined page groups. It is also possible to define an *un-named* ENTRY=DMCL statement, which acts as a *catchall* for any DMCL which may be used.

The following illustrates valid combinations of #COIMAGE macros:

```
#COIMAGE    ENTRY=DMCL,....
#COIMAGE    ENTRY=PG,.....
#COIMAGE    ENTRY=PG,....
#COIMAGE    ENTRY=LAST
```

The above defines a catchall set of EZ-Image file definitions for any run-time DMCL.

```
#COIMAGE    ENTRY=DMCL,NAME='GLB1DMCL',....
#COIMAGE    ENTRY=PG,.....
#COIMAGE    ENTRY=PG,....
#COIMAGE    ENTRY=DMCL,NAME='GLB2DMCL',....
#COIMAGE    ENTRY=PG,.....
#COIMAGE    ENTRY=PG,....
#COIMAGE    ENTRY=LAST
```

The above COIMAGE table defines alternative EZ-Image file definitions dependent on which DMCL is used at run-time.

The following sections define the specific COIMAGE table parameters in full.

4.2.1 #COIMAGE ENTRY=DMCL

```
#COIMAGE ENTRY=DMCL,         ( DMCL | PG | LAST )
       NAME=,                ( DMCL NAME )
       MODE=SYSTEM,          ( CV | LOCAL | SYSTEM )
       MESSAGES=YES,         ( YES | NO )
       DATABUF=100,          ( # DATA BUFFERS)
       INDXBUF=150,          ( # INDEX BUFFERS)
       KEYLEN=6              ( MAX KEY LENGTH )
       SHRPOOL=0             ( VSAM SHRPOOL number)
```

Figure 7 #COIMAGE ENTRY=DMCL Syntax.

Figure 7 shows the full syntax of the COIMAGE ENTRY=DMCL macro. Each of the available parameters is described below:

NAME: You may specify NAME=***dmcl-name***, where ***dmcl-name*** must be a 1-8 character name matching a valid CA-IDMS DMCL name.

If NAME= is not specified, this ENTRY=DMCL definition will act as a *catchall.*

If you are using a *catch-all* DMCL entry, you should ensure that it is always placed last in the COIMAGE table. This is because EZ-Image will search the COIMAGE table from first to last until it finds a match on DMCL name and Mode - If the catchall is the first entry, it will always match on it, and the remaining entries will never be checked.

At run-time, EZ-Image will attempt to locate a COIMAGE ENTRY=DMCL definition for the run-time DMCL module. If not found, and a catchall entry is defined, that entry will be used. If neither a matching entry nor a catchall is defined, then EZ-Image will be disabled.

NOTE: It is also possible to activate EZ-IMAGE using the run-time parameters.

MODE: This parameter allows you to specify different definitions for the same DMCL depending on whether the current job is a Central version or local mode batch job.

SYSTEM: This is the default value of this parameter. This means that you wish to use this DMCL definition in either CV or local mode environments. If you specify (or default to) MODE=SYSTEM, you must only include one occurrence of this DMCL in the COIMAGE table.

CV: Specify MODE=CV if you wish to use this DMCL definition only when using this DMCL in CV processing environment.

LOCAL: Specify MODE=LOCAL if you wish to use this DMCL definition when using this DMCL in local mode processing environment only.

MESSAGES: This parameter is used to enable or disable the Information messages written by EZ-Image to the MVS job log, when using this DMCL.

YES: Enable messages.

NO: Disable messages.

DATABUF: This parameter is used to specify how many VSAM buffers are to be allocated for the data portion of the VSAM records. The higher the number of buffers, the greater hit rate of the image VSAM records you are likely to get. The default value is 100.

INDXBUF: This parameter is used to specify how many VSAM buffers are to be allocated for the index portion of the VSAM records. The default value is 150.

KEYLEN: This is required when the native VSAM files have a KEYLEN greater than the default EZ-Image value of 6.

SHRPOOL: This parameter is used to set the SHRPOOL number used when the VSAM Buffer Pool is built. This would be done to avoid conflicts with CA-IDMS when the DMCL contains a native VSAM area. Default value is 0.

When an EZ-Image database is defined, and the underlying database contains a native VSAM area, unpredictable errors can occur, including CV ABEND 3964, error status nn70, OPEN error messages IEC161I 120-053, IEC161I 156-053, ACB feedback codes 216 & 220.

The problem arises due to the way the VSAM LSR buffer pools are built by CA-IDMS and EZ-Image. Without EZ-Image installed, the VSAM LSR buffers are built by IDMS using the default SHRPOOL number of 0. When installed, EZ-Image will get in first and build the buffers itself using this same default and specifying the CISIZEs and KEYSIZE from it's own files. When the VSAM files are opened, CA-IDMS will then share this same LSR pool and this is where conflicts can occur with key length or CI size. The SHRPOOL parameter works by changing EZ-Image's default number of 0 thus forcing EZ-Image and IDMS Native Vsam files to use separate LSR pools and so sidestep potential conflicts.

SHRPOOL is a parameter on the BLDVRP & GENCB IBM macros. See the appropriate IBM documentation for more details.

4.2.2 #COIMAGE ENTRY=PG

```
#COIMAGE ENTRY=PG,         ( DMCL | PG | LAST )
         IDMSPG=0,         (IDMS BASE PAGE GROUP)
         IMAGEPG=0,        (EZ-IMAGE PAGE GROUP)
         IDMSSEG=' ',      (IDMS SEGMENT NAME)
         IMAGSEG=' ',      (EZ-IMAGE SEGMENT NAME)
         LOPAGE=0,         (LOW PAGE NUMBER IN PAGE GROUP)
         HIPAGE=,          (HI PAGE NUMBER IN PAGE GROUP)
         COMPRESS=FREE,    (FREE | MAX )
         IGNPGLK=YES,      (IGNORE PAGE LOCK ON IDMS FILE)
         DDNAME=,          (EZ-IMAGE VSAM FILE NAME(S))
         BACKUP=NO,        (YES | NO ),
         DUPLEX=NO,        (YES | NO ),
         SCAN=B,           (B(ACK) | F(ORWARD))
         FILEMSK='??????@@',      (MASK FOR NEW IDMS DDNAME)
         FNAMMSK='??????????????@@', (MASK FOR FILE NAME)
         FILEKEY=' ',             (KEY FOR FILE MASK)
         AREAMSK='???????????-PG@@', (MASK FOR AREA NAME)
         AREAKEY=' '              (KEY FOR AREA MASK)
         DBHOT=DBHOT,      (HOT BACKUP TASK CODE)
         SETIMG=UPDATE,
         SETIDMS=DMCL
```

Figure 8 #COIMAGE ENTRY=PG Syntax.

Figure 8 shows the full syntax of the COIMAGE ENTRY=PG macro. each of the available parameters is described below.

IDMSPG This parameter is used to define the Page Group of the underlying (base) database for which you wish to create a EZ-Image (virtual) database. The default for this parameter is 0, for page group zero. Bear in mind that you may create virtual images of production and test or development databases.

IMAGEPG This parameter is used to define the page-group to be used for the EZ-Image virtual database corresponding to the database defined by IDMSPG. You must assign a unique IMAGEPG for each virtual image of the same underlying database. You may define multiple images of the same underlying database by including multiple ENTRY=PG statements in the same COIMAGE table.

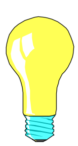
EZ-Image will automatically generate the necessary Subschema tables and DMCL changes required to access the EZ-Image database. The EZ-Image File and Area names are generated from the underlying File and Area names and a replace pattern defined by the FILEMSK and AREAMSK parameters defined later.

Note that the IMAGEPG value must be not be less than the IDMSPG value.

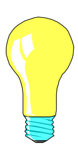
IDMSSEG This is for use in CA-IDMS Release 12.0 environments only. This parameter defines the underlying CA-IDMS segment name that you wish to create a EZ-Image database for.

IMAGSEG This is for use in CA-IDMS Release 12.0 environments only. This parameter defines the segment name to be used for the EZ-Image virtual database, corresponding to the database defined by IDMSSEG. You must assign a unique IMAGSEG name to each virtual image of the same underlying database.

For CA-IDMS Release 12.0 systems, you must specify BOTH the segment names and the page groups of both the underlying base CA-IDMS database and the new EZ-Image virtual database.



If you do not want to have EZ-Image automatically expand the DMCL during startup, you may if you wish add all the required area and file statements to your DMCL and generate a new DMCL. The new files should map to the physical files that you wish to image. You should then create a COIMAGE table with a PG entry where IMAGPG and IDMSPG both point at the new Imaged page group you have defined in the DMCL.

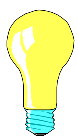


For example, suppose your existing database is page group10, and you want to create a EZ-Image page group 20, which is an imaged copy of page group 10.

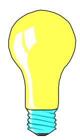
You could add all the required area and file statements for page group 20 to the DMCL and regenerate it. All the page group 20 files would be mapped to the page group 10 physical files. Then create a COIMAGE table with a PG entry specifying IDMSPG=20,IMAGEPG=20. EZ-Image would then create the image page group 20 based upon the definition you have added to the DMCL, and would use the page group 20 files as the underlying physical database.

LOPAGE=0

HIPAGE= ***10.2 Only;*** These parameters may be used if you wish EZ-Image to create a virtual image of only a subset of the areas in the applicable page group. By default, EZ-Image will create a virtual image of the whole IDMS page group. If you specify LOPAGE without specifying HIPAGE, then HIPAGE will default to the highest available page in the page group. Similarly, you may specify HIPAGE without LOPAGE, in which case LOPAGE will default to zero.



For example, in a production environment where everything is page group zero, and you do not wish to image the dictionary areas, you may specify LOPAGE=nnnnnn, where nnnnnn is the starting page number of the 1st non-dictionary area that you wish to image.



In 12.0, users can exclude areas that they do not want to image by moving them into a new segment.

COMPRESS EZ-Image will compress data stored in the EZ-Image VSAM file using one of three methods.

FREE Remove free space in the middle of the CA-IDMS page before storing in the EZ-Image file. This is the default value.

MAX Remove Free Space and repeating characters from the CA-IDMS page, by checking for repeating full words, before storing in the EZ-Image file. This option reduces the space requirement for the EZ-Image file, but is more costly in terms of CPU.

IGNPGLK If the underlying database is available for update to the CA-IDMS system- for example in a system where you are imaging a production database, you should specify IGNPGLK=YES to allow EZ-IMAGE to open the virtual image of this file in update mode also.

NOTE: Under NO circumstances are any updates written back to the underlying database by a user running against the virtual image of the file.

DDNAME Specify DDNAME=**ddname,** where **ddname** is the name of the VSAM file to be used for this virtual database. The VSAM file must be allocated and initialized as described later in this document, and must be included in the applicable run-time JCL.

The maximum size of a VSAM file is 4 Gigabytes, which will usually be more than adequate to hold the contents of a EZ-Image database. However, if you anticipate exceeding this capacity, it is possible to allocate a EZ-Image database across multiple VSAM files by specifying this parameter as follows:

DDNAME=(ddn$_1$,hipage$_1$,ddn$_2$,hipage$_2$,....ddn$_n$,hipage$_n$)

where ddn$_1$ thru ddn$_n$ define the names of the VSAM files to be allocated for this EZ-Image database, and hipage$_1$ thru hipage$_n$ define the highest page number to be allocated to the preceding VSAM file name.

NOTE - The maximum number of VSAM files that an Image Page Group can be split across, is eight.

For example, if you specify DDNAME=(IMAGE01,500000,IMAGE02) then 500000 will be the highest page number stored in IMAGE01. The remaining page range from 500001 to the highest page in the database will be stored in IMAGE02.

BACKUP Identifies whether this definition is for a Hot-backup entry.

YES: This is a Hot Backup definition.

NO: This is not a Hot Backup definition. This is the default value.

DUPLEX Identifies whether this definition is for a Duplex entry.

YES: This is a Duplex definition.

NO: This is not a Duplex definition. This is the default value.

Refer to **section 6** for further information.

DBHOT This parameter is used to define the task code which is used to STOP and START a EZ-Image “Hot Backup”. The default value is DBHOT.

Refer to **section 6** for further information.

SETIMG This parameter is used to specify the startup mode of the *image* database. The parameter may have the following values:

UPDATE:	Set the open mode to Update.
RETRIEVAL:	Set the open mode to Retrieval.
TRANSIENT:	Set the open mode to Transient Retrieval.
OFFLINE:	Set the open mode to Offline.

The default value of this parameter is UPDATE. Refer to the CA-IDMS manuals **System Tasks and Operator Commands** and **Database Administration** for a full description of these parameters.

SETIDMS This parameter is used to specify or override the startup mode of the underlying (base) database. If specified, this will override the ON STARTUP SET STATUS TO segment specification of the CA-IDMS DMCL. The parameter may have the following values:

DMCL:	Set the Open mode of the underlying database according to the DMCL definition.
UPDATE:	Set the open mode to Update.
RETRIEVAL:	Set the open mode to Retrieval.
TRANSIENT:	Set the open mode to Transient Retrieval.
OFFLINE:	Set the open mode to Offline.

The default value of this parameter is DMCL. Refer to the CA-IDMS manuals **System**

Tasks and Operator Commands and **Database Administration** for a full description of these parameters.

The remaining parameters on the ENTRY=PG type statement are for the generation of the Image Area and File names. These parameters are discussed in the following section.

4.2.3 Generation of EZ-Image File and Area names.

SCAN This parameter is used to determine whether the CA-IDMS area and/or file name is scanned forwards (F) or backwards (B) when searching for the FILEKEY or AREAKEY as described later in this section. The default is backwards (B).

FILEMSK='??????@@'
FNAMMSK='??????????????@@'
FILEKEY=' '
AREAMSK='??????????-PG@@'
AREAKEY=' '

EZ-Image automatically generates an expanded DMCL including area and file definitions required for the EZ-Image database(s). In order to do this, EZ-Image must generate unique area names and internal and external file names for each area and file in the EZ-Image databases.

Though the names of the generated areas and files do not matter, so long as they are unique, users may wish to generate names that conform to local conventions. These parameters are available to allow the user to define how the names are to be built.

Use of the following parameters is best described through example. For the sake of the description, we will assume that the base database is page group zero, and the EZ-Image virtual database is to be assigned to page group 50. Therefore, your COIMAGE table will include the parameters

```
#COIMAGE ENTRY=PG,
      IDMSPG=0,
      IMAGEPG=50,
      .
      .
```

In the following parameters, special characters may be included as follows: The question mark (?) symbol means - substitute the corresponding character in the IDMS file or area name in the EZ-Image file or area name.

Any other character will be inserted at the equivalent position, and embedded blanks removed.

The @ symbol is a special character representing the EZ-Image page group number. The @ character(s) will be replaced by the EZ-Image page group number.

FNAMMSK='??????????????@@'

This parameter is used to generate a new internal file name for the EZ-Image file.

For example, if you specify

FNAMMSK='???????????-IM@@'

then IDMS file name PROD-DF001A would become PROD-DF001A-IM50, where the last five characters have been replaced with '-IM50', where 50 is the EZ-Image page group.

FILEMASK='??????@@'

This parameter is used to generate a new external (DDname) for the EZ-Image file.

For example, if you specify

FILEMASK='??????@@'

then IDMS file DF001A would generate a DDname of DF001A50 for the EZ-Image file, where 50 is the EZ-Image page group.

FILEMASK='DK??????'

would generate a name DDname of DK001A for the EZ-Image file, the first two characters having been replaced with DK. This time no page group number has been added as the mask did not contain any @ characters.

FILEKEY=' '

You may specify this parameter if you wish to search the File name for a particular character string to be replaced by the mask as described above.

For example, if your site standard includes the page group identifier in the file name, you may specify

FILEKEY=00

in which case IDMS Ddname PKLUP00 would generate a Ddname of PKLUP50 for the EZ-Image file.

AREAMSK='???????????-PG@@'

This parameter is used to generate the new area names for the EZ-Image database.

For example, taking the default value of this parameter,

IDMS area SECURE-AREA would generate an area name of SECURE-AREA-PG50 for the corresponding EZ-Image database area.

AREAKEY=' '

You may specify this parameter if you wish to search the IDMS area name for a particular character string to be replaced by the mask as described above.

For example if you specify

```
AREAKEY='-AREA'
AREAMSK='???????????-AR@@'
```

then IDMS area name DB001-SECUR-AREA would be generated as the EZ-Image database area name DB001-SECUR-AR50.

Note: The character string AREAKEY must be the same length as the replacement character string.

4.2.4 #COIMAGE ENTRY=LAST

```
#COIMAGE      ENTRY=LAST    ( DMCL | PG | LAST )
```

Figure 9 #COIMAGE ENTRY=LAST Syntax.

Figure 9 shows the syntax of the COIMAGE ENTRY=LAST macro.

There must always be one ENTRY=LAST macro when generating your COIMAGE table, and it must be the final entry to signify the end of the table.

4.3 Creating the EZ-Image VSAM files.

The EZ-Image databases are stored in VSAM indexed files. A sample job is provided in the installation source library called **ALOCIMAG** which should be used to allocate and initialize the new EZ-Image VSAM files. The job consists of two steps, namely an IDCAMS step, to delete and create the file, and a REPRO step to initialise it.

Note: This job can be used subsequently to re-initialize a given EZ-Image database when you want to clear down its contents. Since the IDCAMS step starts with a DELETE statement for the file, you will receive a return code of 8 the first time this job is run for a particular file.

A description of the job steps contained in ALOCIMAG are described below:

```
//STEP1  EXEC PGM=IDCAMS,REGION=1536K
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DELETE IMAGE.DSN
  DEFINE CLUSTER( -
                NAME (IMAGE.DSN)                -
                VOL (VVVVVV)                    -
                CYLINDERS ( NN NN )             -
                INDEXED                         -
                REUSE                           -
                SHAREOPTIONS (2 3)              -
                )                               -
           DATA ( -
                NAME (IMAGE.DSN.DATA)           -
                KEYS(6,10)                      -
                RECORDSIZE(AAAAA,MMMMM)         -
                SHAREOPTIONS (2 3)              -
                )                               -
           INDEX ( -
                NAME (IMAGE.DSN.INDEX)          -
                SHAREOPTIONS (2 3)              -
                IMBED                           -
                )
```

Figure 10 Allocating a EZ-Image VSAM dataset.

Figure 10 shows a sample IDCAMS job stream required to allocate a VSAM file to contain a EZ-Image database. You need to allocate at least one VSAM file the EZ-Image databases that you are activating. Note that in most cases, a single VSAM file should be sufficient to hold an entire EZ-Image virtual database, however, it is possible to allocate a particular EZ-Image database across multiple VSAM files, or several EZ-Image page groups to one VSAM file.

Refer to section 4.2 "Generating the COIMAGE table" for further details.

In the IDCAMS job, you must modify the supplied job-stream as follows:

IMAGE.DSN Change this to the dataset name to which you wish to assign this particular EZ-Image VSAM file.

VVVVVV Change this to the VOLUME on which this file will be allocated.

NNNN Change the primary and secondary allocations on the CYLINDERS parameter to the space required for this file.

MMMMM This is the maximum page size. It should be set to the maximum page size in the IDMS DMCL + 16 bytes.

AAAAA This is the average page size in the file after compression. This refers to the actual space used in the IDMS database. You can size this parameter based upon analysis of IDMSDUMP reports. If unsure, set this parameter to 60% of the maximum size (MMMMMM).

The following shows sample JCL for the REPRO step required to initialize the previously defined VSAM file.

This step must be run after the allocation of every new EZ-Image VSAM file, before it can be used by EZ-Image.

```
//REPRO EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//INDD1    DD DSN=EZ-DB.SRCLIB(DBIMINIT),DISP=SHR
//OUTD1    DD DSN=IMAGE.DSN,DISP=SHR
//SYSIN    DD *
   REPRO INFILE(INDD1) -
         OUTFILE(OUTD1)
/*
```

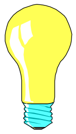
Figure 11 Initializing a EZ-Image VSAM database.

The JCL in the REPRO step shown in **Figure 11** should be modified as follows:

EZ-DB.SRCLIB This is the name of the library to which you unloaded the EZ-DB installation tape source.

IMAGE.DSN This is the dataset name of the VSAM file as specified in the previously described IDCAMS step.

4.4 Generating the new Subschema tables (10.2 only)



It is not necessary to generate new subschemas for EZ-Image in CA-IDMS Release 12.0 environments. The same subschemas can be used for both the underlying physical databases and the virtual image databases.

4.4.1 Running COIMZAPS

A program is provided called COIMZAPS whose function is to generate IEBCOPY and AMASZAP parameters to copy and create the new subschema load modules required to access the EZ-Image database.

Skeleton JCL required to execute COIMZAPS is provided on the installation tape source library. An example of JCL for COIMZAPS and its use is described below.

```
//RUNZAPS  EXEC PGM=COIMZAPS
//STEPLIB  DD   DSN=EZ-DB.loadlib,DISP=SHR
//SYSIN    DD   *
IDMSPG=02 IMAGEPG=12  SUBSKEY=SS02 SUBSMASK=???SS@@?
/*
//SUBSZAPS DD DSN=punch.data(zapparms),DISP=SHR
//SUBSCOPY DD DSN=punch.data(copyprms),DISP=SHR
//SYSUT1   DD DSN=&&SYSUT1,UNIT=VIO,SPACE=(800,(20,20)),
//            DISP=(,DELETE),
//            DCB=(RECFM=FB,BLKSIZE=800,LRECL=80)
//SUBSIN   DD   *
AAASS02A
BBBSS02A
CCCSS02A
/*
```

Figure 12 Executing the COIMZAPS program.

A sample COIMZAPS job is shown above in **Figure 12.** The JCL should be modified as follows;

EZ-DB.loadlib Is the name of the library containing the EZ-DB runtime modules.

punch.data(zapparms) Is the name of a partitioned dataset member to which the parameters for the subsequent AMASPZAP job are written.

punch.data(copyprms) Is the name of a partitioned dataset member to which the parameters for the subsequent IEBCOPY job are written.

Note: The SYSUT1 work file must be present.

The jobstream contains a SYSIN file in which you specify the subschema conversion parameters. These parameters are used to control how COIMZAPS will modify and name the newly generated subschema tables.

The SYSIN control parameters are described below.

IDMSPG=
IMAGEPG=

Specify the page group of the IDMS database (IDMSPG) and the corresponding page group for the EZ-IMAGE database (IMAGEPG).

SUBSKEY
SUBSMSK
SCAN

These parameters are used to control how the newly generated subschema tables are to be named by modifying the original subschema name. The function of the KEY, MASK and SCAN parameters is the same as that used to control the generation of new FILE and AREA names as described in the section "Generating the COIMAGE table".

For example, taking the first SYSIN input statement in the above example, subschema AAAASS01 would be regenerated with a name of AAAASS99.

For a full description of the use of SUBSKEY, SUBSMASK and SCAN, refer to the COIMAGE table parameters FILEKEY FILEMASK and SCAN as appropriate.

Note: Unlike the COIMAGE table parameters, the SYSIN parameters should not be enclosed in quotes eg. SUBSMASK=???SS@@?

You may include as many statements in the SYSIN file as required, depending on the number of EZ-Image virtual databases you are creating.

SUBSIN

The job-stream contains a list of the subschemas that you require to be regenerated for use with the EZ-Image databases. Each subschema name must start in column one. You can specify as many subschema names as you require in a single run.

4.4.2 COPY the SUBSCHEMAS

After running COIMZAPS, you will have created two parameters file. The first as defined in the *SUBSCOPY* file, should be used as input to an IEBCOPY job.

Figure 13 shows an example of an IEBCOPY job using the output from COIMZAPS as input to the

```
//IEBSCPY1 EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//SYSUT3   DD UNIT=VIO,SPACE=(CYL,1)
//SYSUT4   DD UNIT=VIO,SPACE=(CYL,1)
//INDD     DD DSN=subschema.loadlib,DISP=SHR
//OUTDD    DD DSN=&&SUBS,DISP=(,PASS),SPACE=(CYL,(10,10)),
//            DCB=(BLKSIZE=800,LRECL=80,RECFM=FB)
//SYSIN    DD DSN=punch.data(copyprms),DISP=SHR
//*
//IEBSCPY2 EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//SYSUT3   DD UNIT=VIO,SPACE=(CYL,1)
//SYSUT4   DD UNIT=VIO,SPACE=(CYL,1)
//INDD     DD DSN=&&SUBS,DISP=(OLD,DELETE)
//OUTDD    DD DSN=subschema.loadlib,DISP=SHR
//SYSIN    DD *
 COPY INDD=((INDD,R)),OUTDD=OUTDD
```

Figure 13 Copying the subschemas.

IEBCOPY job. Note that the copy is done in two stages - using a temporary intermediate dataset, as IEBCOPY cannot copy and rename members to the same dataset.

This will copy the required subschemas from their existing library into a new library, and in the process rename them according to the parameters you used as input to the run of COIMZAPS. Note that you must have punched and linkedited the subschemas into a suitable load library before this step can be run.

4.4.3 MODIFY the new SUBSCHEMAS

After the subschemas have been copied and renamed, you should use the SUBSZAPS parameter file created by COIMZAPS, as input to a run of AMASPZAP. This will modify the subschema name and the page group within the subschema load module.
Figure 14 shows an example of an AMASPZAP job that uses the ouput from COIMZAPS as input to

```
//SUBSZAP  EXEC PGM=AMASPZAP
//SYSPRINT DD SYSOUT=*
//SYSLIB   DD DSN=subschema.loadlib,DISP=SHR
//SYSIN    DD DSN=punch.data(zapparms),DISP=SHR
```

Figure 14 Modifying the subschemas.

AMASPZAP.

Each of the new subschemas will be 'zapped' with its new subschema name, and new image page group.

Once the subschemas have been modified, they are ready to be used for accessing the EZ-Image databases.

4.5 Generating the new DBNAME table

In order to access the new image databases you have created, you will need to generate a new DBNAME table, to include details of the new subschema mappings, and in the case of CA-IDMS Release 12, to include details of the EZ-Image segment names. Check your CA-IDMS documentation for full details of generating a new DBNAME Table.

Run Time JCL Requirements

5 Run Time JCL Requirements

5.1 Modifying the Job or Startup JCL

In order execute a job using EZ-Image, you must include various datasets and control files in the job-stream of the CV startup or local mode batch JCL, as described in this section.

Figure 15 shows an example of a CV startup job-stream modified to include EZ-Image database processing.

```
//IDMSDC    EXEC PGM=IDMSDCV,
//CDMSLIB   DD   DSN=EZ-DB.LOADLIB,DISP=SHR
//          DD   DSN=EZ-DB.IMAGE.LOADLIB,DISP=SHR
//DBIMCTL   DD   DSN=EZ-DB.DBIMCTL,DISP=SHR
//DBIMLOG   DD   SYSOUT=*
```

Figure 15 Modifying the run-time JCL.

Note the following JCL statements.

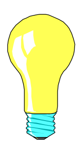
CDMSLIB The CDMSLIB concatenation has been modified to include the EZ-DB.LOADLIB, that is, the library containing the EZ-DB and EZ-Image run-time modules, and EZ-DB.IMAGE.LOADLIB, which is the library containing the regenerated EZ-Image subschemas and applicable COIMAGE table if being used.

Note: You may, if you wish, include the subschema table and COIMAGE tables in the EZ-DB.loadlib if you prefer.

DBIMCTL This control file is optional as described later. If you do not include a DBIMCTL file, then EZ-Image will default to using the table COIMAGE, unless you have specified an alternative name through the //COXACTL file COIMAGE= parameter.

DBIMLOG Specify the name of the dataset to which the EZ-Image shutdown stats will be written.

You can if you wish specify SYSOUT= for this parameter.



It is NOT necessary to include the generated IDMS file names for the EZ-Image database files or the EZ-Image VSAM files in the run-time JCL. These are dynamically allocated by the DB-Allocate component of EZ-DB.

5.2 DBIMCTL parameters

It is possible to control the EZ-Image run-time environment through the use of an input control file in the run time JCL. If present the DBIMCTL will override any COIMAGE table that has been created.

The run-time control file should be named DBIMCTL. The permitted parameters may be included in-stream as shown below:

```
//DBIMCTL   DD    *
<parameter>
<parameter>
```

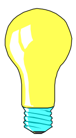
or alternatively, you may specify the parameters through an input dataset, eg.

```
//DBIMCTL   DD  DSN=parameter.file,DISP=SHR
```

where *parameter.file* is allocated as RECFM=F or FB, and LRECL=80.

The permitted parameters which may be input through DBIMCTL are equivalent to those allowed in the generation of the COIMAGE table ENTRY=PG syntax. You should refer to section 4.2 - "Generating the COIMAGE table" for a description of the parameters.

Since the parameters will always relate to the applicable run-time DMCL for the job, there is no equivalent of the ENTRY=DMCL syntax.



If no COIMAGE table has been defined, the standard defaults will be in effect for the current DMCL.

There are two parameters available to override the default VSAM buffer specifications:

```
DATABUF=
INDXBUF=
```

These can be appended to any DBIMCTL page group specification, and need only be specified once. Refer to section 4.2.1 for details.

There are two additional DBIMCTL parameters, namely :-

```
REORG=YES|NO
IDMSUTIL=YES|NO
```

The use of these parameters is described later in Technical Notes (**section 7**).

The following figure (**Figure 16)** shows an example of the use of DBIMCTL:-

```
//DBIMCTL  DD *
* image page group 66
IDMSPG=52 IMAGEPG=66 AREAKEY=-AR52 FILEKEY=52                X
DDN=(COPG6601,11546600,COPG6602)                             X
LOPAGE=8500000                                               X
AREAMSK=??????????-AR@@
*image page group 70                                         X
IDMSPG=52 IMAGEPG=70
DDN= etc etc.
*
```

Figure 16 Example of DBIMCTL file usage.

The rules relating to the specification of DBIMCTL statements are as follows:-

Comments are permitted on separate cards or anywhere on the line, indicated by an asterisk character.

All parameter keywords may be abbreviated down to the minimum number of characters which make that keyword unique.

Parameters may start in any column (1-71). Any number of spaces (not commas) are permitted between parameters. Column 72 is treated as a continuation character, but note that a single keyword and its parameter may not span cards.

5.3 DBIMLOG

The **DBIMLOG** control statement should be included if you wish EZ-Image to write statistics at job termination or CV shutdown.

Details of the output is given in section 3.1 - Shutdown Reports.

EZ-HotBackup and EZ-Duplex Options

6 EZ-HotBackup and EZ-Duplex Options

The implementation of EZ-Image lends itself very effectively to provide ***Hot Backup*** and ***Duplex Database*** functionality to CA-IDMS users, in a very simple and cost effective manner compared to the traditional way of implementing these functions.

The way these facilities are provided through EZ-Image is documented in this section.

6.1 *EZ-HotBackup* Overview

Hot Backup is the ability to take a *backup* or *secure copy* of a CA-IDMS database while that database is still available for update to the end-users. It is common practise for CA-IDMS users to take *static back-ups* of their database areas (that is to say, with the applicable database areas varied offline or to retrieval). This is because the standard procedure available within CA-IDMS to perform a Hot Backup is considered to be over complicated or un-reliable. For sites wishing to offer 24-hour (or simply to extend) availability to their end-users, the static backup procedure is a major limitation.

Fortunately, for EZ-Image users these problems become a thing of the past. EZ-HotBackup makes taking a Hot Backup as simple as would be the case when taking a static backup. The way this is implemented is to associate with a CA-IDMS database a EZ-Image image file to be used during Hot Backup processing. When the Hot Backup procedure is started, EZ-Image ensures that updates to the underlying CA-IDMS database are written to the image file only. The backup of the underlying database therefore takes place as though it were a normal static backup, since the underlying database is in fact not being modified. When the backup is completed, a EZ-Image procedure is invoked to copy the updates from the Hot-Backup image to the underlying database.

The definition of the Hot-backup Image files and the operational procedure are now described later in this section.

6.2 EZ-Duplex Overview

The concept of a Duplex database is to provide increased system resilience and reduce system down-time associated with physical device errors. Typically this is achieved by replicating the entire database (or the subset for which you require duplication) and using the IDMSDPLX exit to mirror all updates to the prime database against the duplex (mirror) database. In the event of a physical error on one of the files being duplexed, recovery time is minimized because you can simply switch the files using dynamic allocation, or shutdown and warmstart using the duplex file in place of the original file.

Though highly desirable, the above scenario is implemented by only a small minority of CA-IDMS users. The complex procedures and high cost of DASD make it very rare that a user can cost-justify such an implementation.

However, with EZ-Image, it is possible to implement a form of database duplication that satisfies the basic objective of duplexing, that is minimized recovery time, in a simple and more cost effective manner. The way this is implemented is to associate with a CA-IDMS database a EZ-Image image file to be used as a Duplex file. When a CA-IDMS file has a Duplex image assigned, EZ-Image ensures that updates to the underlying CA-IDMS database are written to the base CA-IDMS file AND the duplex image file. The Duplex *image* requires less physical DASD than a *real* Duplex database, because the *duplex image* contains only modified pages (and in compressed format).

Database backups of the base CA-IDMS database are carried out exactly as before. In the event of a physical failure, the user need only to restore the damaged file from the last backup, and use the DBIMUTIL UPDIDMS function to apply the updates from the duplex image to the restored database. There is no need to run IDMSRFWD to roll-forward from the off loaded journals (including the necessary journal file concatenation). The most time consuming part of the database recovery is therefore not required.

6.3 Hot Backup and Duplex Definition

6.3.1 Generating the COIMAGE table

Hot-Backup and Duplex database *images* are defined using the COIMAGE table (or DBIMCTL parameters) in much the same way as normal *images* are defined. This section describes the parameters as applicable to Hot Backup and Duplex database *images*. For a full description of the COIMAGE table refer to section 4.2.

A Hot-Backup or Duplex image is defined using #COIMAGE ENTRY=PG statement. The required parameters are as follows:-

```
#COIMAGE ENTRY=PG,
      BACKUP=YES,             ( Defines this as a HOTBKUP image )
      DUPLEX=YES,             ( Defines this as a DUPLEX image )
      IDMSPG=page-group       ( IDMS base page  g r o u p )
      IDMSSEG=seg-name        ( IDMS segment name)
      LOPAGE=0,               ( Low page number in page group)
      HIPAGE=,                ( Hi page number in page group)
      COMPRESS=FREE,          ( FREE | MAX)
      DDNAME=                 ( VSAM file names(s) )
```

The parameters BACKUP and DUPLEX are mutually exclusive. That is, a particular CA-IDMS file cannot be duplexed and also eligible for Hot-Backup.

The following considerations apply depending on whether you are running CA-IDMS release 10.2 or 12.x.

CA-IDMS 10.2 Users;

You must include one ENTRY=PG statement for each and every base CA-IDMS page-group on which you wish to perform Hot Backups or Duplex. For example, the following might be a typical release 10.2 definition:

```
#COIMAGE ENTRY=DMCL,NAME=GLBLDMCL
#COIMAGE ENTRY=PG,BACKUP=YES,IDMSPG=0,DDNAME=HOTBK00
#COIMAGE ENTRY=PG,DUPLEX=YES,IDMSPG=10,DDNAME=DUPLX10
#COIMAGE ENTRY=PG,BACKUP=YES,IDMSPG=20,DDNAME=HOTBK20
#COIMAGE ENTRY=LAST
```

In this example, IDMS page groups 0 and 20 both have Hot-Backup images defined. IDMS page group 10 is being duplexed.

CA-IDMS 12.x Users;

You must include one ENTRY=PG statement for each and every base CA-IDMS Segment on which you wish to perform Hot Backups or Duplex. If a segment contains more than one page-group, you must include an ENTRY=PG statement for each page group in the segment. For example, the following might be a typical release 12.0 definition:

```
#COIMAGE ENTRY=DMCL,NAME=R120DMCL
#COIMAGE ENTRY=PG,BACKUP=YES,SEGMENT=SYSDICT,
    DDNAME=BKUPSYS
#COIMAGE ENTRY=PG,BACKUP=YES,SEGMENT=EMPLSEG,
    IDMSPG=0,DDNAME=BKUPEMPL
#COIMAGE ENTRY=PG,BACKUP=YES,SEGMENT=EMPLSEG,
    IDMSPG=50,DDNAME=BKUPEMPL
#COIMAGE ENTRY=PG,DUPLEX=YES,SEGMENT=APPLSEG,
    DDNAME=DUPLXAPP
#COIMAGE ENTRY=LAST
```

This example illustrates how you would code Hot-Backup definitions for a segment (EMPLSEG) with multiple page-groups.

6.3.2 Defining the Image VSAM Files

The VSAM files corresponding to the Hot-Backup and Duplex database images must be allocated and initialized in the same manner as normal image databases. A skeleton JCL member ALOCIMAG is included in the installation source library for this purpose. A full description of how to allocate and initialise the required VSAM files can be found in **section 4.3**.

Note: It is not necessary to include DD statements for the VSAM files in the CA-IDMS run-time JCL as they can be allocated dynamically by the DB-Allocate component of EZ-DB. In this case, the names must be defined in the DB-Allocate COALLOC file. Refer to the DB-Allocate user guide for further information.

6.4 Hot Backup Operational Procedure

Once a Hot-Backup Image has been defined for a particular CA-IDMS CV, it can be activated and de-activated using the DBHOT START and DBHOT STOP commands. The following describes the way these commands and the Hot-Backup image file(s) are used to facilitate Hot-Backups of CA-IDMs databases.

To take a Hot-backup of a particular CA-IDMS database proceed as follows:-

1) Vary applicable database areas to retrieval.

For example using DCMT VARY AREA or DCMT VARY SEGMENT commands. Refer to **CA-IDMS System Task and Operator Commands** for further details.

2) Activate the EZ-Image Hot-Backup file(s).

Once the applicable Hot-Backup image files have been activated, any updates performed against the underlying CA-IDMS files are written to the image files only.

The Hot-backup images are activated by issuing the command **DBHOT START** from a DC/UCF *enter next* *task code* prompt as follows:- .

DBHOT START PG=*page-group*	(CA-IDMS 10.2 systems)
DBHOT START SEGMENT=*segment-name*	(CA-IDMS 12.0 systems)

where *page-group* is the page-group of the applicable CA-IDMS database to be backed up (10.2 system) and *segment-name* is the segment-name of the applicable CA-IDMS database to be backed up. You must issue this command for each applicable segment or page-group.

The following message will be written to the system log:-

12.0	**DB905611 EZ-DB HOT BACKUP STARTED IDMS PG** *pgrp* **SEGMENT** *segname*
10.2	**DB905611 EZ-DB HOT BACKUP STARTED IDMS PG** *pgrp*

An image file must have been defined for each applicable page-group in the COIMAGE table (or through the DBIMCTL parameter file).

3) Vary the applicable areas back to Update.

Refer to the **CA-IDMS System Task and Operator Commands** manual for further details.

4) Start the database backup.

Use whatever mechanism you currently use for taking your database backups. The database is currently available in Update mode to end-users, but since any updates issued against the underlying database are being written to the Hot-Backup image only, this backup is effectively a *static* backup.

5) When the backup is complete, deactivate the hot-backup image.

The Hot-backup images are de-activated by issuing the command DBHOT STOP from a DC/UCF *enter next task code* prompt as follows:- .

DBHOT STOP PG=*page-group*	(CA-IDMS 10.2 systems)
DBHOT STOP SEGMENT=*segment-name*	(CA-IDMS 12.0 systems)

where *page-group* is the page-group of the applicable CA-IDMS database to be backed up (10.2 system) and *segment-name* is the segment-name of the applicable CA-IDMS database just backed up. You must issue this command for each applicable segment or page-group.

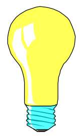
The following messages will be written to the system log for each *segment* and/or *page-group* as applicable:-

12.0	**DB905611 EZ-DB HOT BACKUP STOPPED IDMS PG** *pgrp* **SEGMENT** *segname*
10.2	**DB905611 EZ-DB HOT BACKUP STOPPED IDMS PG** *pgrp*

Followed by the message....

DB905612 EZ-DB HOT BACKUP WROTE BACK PAGES *nnnn*

where **nnnn** is the number of pages that EZ-Image has copied back from the Hot-Backup image file to the underlying database for the corresponding segment/page-group combination.



The copying of the database pages from the Hot-backup image back to the base database happens completely transparently to end-users on the production system. Refer to Technical notes section *Hot Backup questions and answers* for further information.

6.5 Duplex Database Operational procedure

Once defined in the COIMAGE table (or DBIMCTL parameters) there is no user action required to activate the Duplex Image. The image is automatically activated when the system starts up. You can use the DBIM online task to observe the duplex images and monitor how much data has been written to each duplex file.

6.5.1 Backup Procedure using Duplex Images

- In order to co-ordinate the Duplex Images with the backups, it is a requirement that you perform **static backups** of the files which are to be duplexed. You cannot mix Hot-backups with EZ-Image Duplexing.

- On completion of the static backup, before varying the applicable files back to update, you should initialize the corresponding Duplex *image* files.

  If the Central version owning the Duplex image is still running, issue the following command to close the VSAM file:

  DBIMV VARY VSAM=*vsfile* CLOSE

  NOTE: This command can also be input from the DBIM online display command line.

  Then use the DBIMUTIL utility program to initialize the VSAM file as follows:

  INIT DDNAME=*vsfile*

  A full description of the DBIMV functions and DBIMUTIL utility can be found in section 3.

6.5.2 Recovering from failure using a Duplex Image

In the event that you experience a failure of the underlying database file, you should recover the file as follows:-

- Vary the Problem area Offline (if the CV is still running)

- Recover the underlying database using your last backup.

- Copy the updates from the Duplex image file to the restored CA-IDMS database file using the DBIMUTIL UPDIDMS facility as follows:

  UPDIDMS IMAGEPG=nn IDMSPG=nn DMCL=*dmclname*

  where in this case IMAGEPG and IDMSPG will be the same. Refer to section 3.5.4 For a full description of the DBIMUTIL program.

- Restart the CV, or Vary the area back online. In the event of a CV re-start, normal Warmstart processing will take place (and back-out any updates belonging to incomplete rununits).

6.6 EZ-HotBackup and EZ-Duplex Online Displays

The various screens available through the DBIM online task have been enhanced to allow you to monitor the status of Hot-Backup and Duplex Images. The online displays are fully documented in section 3. In this section, the intention is to highly pertinent information as it relates to Hot-Backup or Duplex option.

```
COGITO EZ-IMAGE R4.2                                       V0005  16:55:50 94.354
                *** EZ-IMAGE FILE SUMMARY DISPLAY ***
EZ-IMAGE Table : COIMCTL    # Data Buffs     100  # IMAGE VSAM Files :    3
DMCL Name      : DMGLBD2    # Indx Buffs     150  # IMAGE IDMS Files :   17
                                                  # IMAGE PGs        :    6
   -----Image----  -----IDMS-----  VSAM              ---VSAM Totals---
   File        PG  File        PG  DDNAME    Sta Comp   #Reads  #Writes
 --------------------------------------------------------------------------
            DUPLX  DICTDB       0  COIMPG50  ACT FREE        1        1
            DUPLX  DLODDB       0  COIMPG50  ACT FREE        2        1
   ACTIVE   HOTBK  EMPDEMO      0  COIMPG40  ACT FREE        0        0
   EMPDEM30    30  EMPDEMO      0  COIMPG30  ACT FREE        0        0
   EMPDEM31    31  EMPDEMO      0  COIMPG30  ACT FREE        1        1
   EMPDEM32    32  EMPDEMO      0  COIMPG30  ACT FREE        1        1
   EMPDEM40    40  EMPDEMO      0  COIMPG30  ACT FREE        1        1
   ACTIVE   HOTBK  INSDEMO      0  COIMPG40  ACT FREE        0        0
   INSDEM30    30  INSDEMO      0  COIMPG30  ACT FREE        1        1

 --------------------------------------------------------------------------
   Totals                                                   24       14

<CLEAR> EXIT   <PF7> UP   <PF8> DOWN   <PF12> MODE
ENTER COMMAND BELOW>
```

Figure 17 File Summary Screen - Showing Hot Backup and Duplex Images

Figure 17 shows an example of the DBIM File Summary screen where Hot-Backup and Duplex images are in use. Note, in this example, that the CA-IDMS files DICTDB and DLODDB are being duplexed. This is indicated by the literal **DPLX** which appears in the **Image Page Group** column. For a Duplex image, **Image File** will always be blank as it does not apply. **VSAM DDName** identifies the VSAM file which is used to maintain the Duplex image.

This example also illustrates two CA-IDMS files, namely EMPDEMO and INSDEMO which have Hot-Backup images defined. This is indicated by the literal **HOTBK** which appears in the **Image Page group** column. For a Hot-Backup image, **Image file** will be blank unless a hot-backup is currently in progress (ie. The user has issued the DBHOT START command for this file). In this case, the **Image File** column contains the literal **ACTIVE**.

Figure 18 show an example of the DBIM Page Group Summary screen with Hot Backup and Duplex Images defined. As can be seen for a Duplex or Hot-Backup image, the Image Page Group and Segment do not apply, and these field simply contains the literal DUPLX or HOTBK as applicable.

```
COGITO EZ-IMAGE R4.2                                  V0005  16:55:58 96.042
                *** EZ-IMAGE PAGE GROUP DISPLAY  ***
EZ-IMAGE Table : COIMCTL    # Data Buffs     100  # IMAGE VSAM Files :     3
DMCL Name      : R120DMCL   # Indx Buffs     150  # IMAGE IDMS Files :    17
                                                  # IMAGE PGs        :     6
   -----Image----  -----IDMS-----          #   1st VSAM  ---VSAM Totals---
      PG  Segment     PG  Segment   Comp   DD   DDNAME      #Reads  #Writes
 ---------------------------------------------------------------------------
   DUPLX               0  APPLDICT  FREE    1   COIMPG50         3        2
   HOTBK               0  EMPDEMO   FREE    1   COIMPG40         4        0
      30 EMPIM30       0  EMPDEMO   FREE    1   COIMPG30         3        3
      31 EMPIM31       0  EMPDEMO   FREE    1   COIMPG30         3        3
      32 EMPIM32       0  EMPDEMO   FREE    1   COIMPG30         3        3
      40 EMPIM40       0  EMPDEMO   FREE    1   COIMPG30         6        3




----------------------------------------------------------------------------
   Totals                                                       24       14
<CLEAR> EXIT   <PF7> UP   <PF8> DOWN   <PF12> MODE
ENTER COMMAND BELOW>
```

Figure 18 Page Group Summary Screen - Showing Hot Backup and Duplex Images

EZ-Extract Utility

7 EZ-Extract Utility

7.1 Overview

It is a common requirement for users to validate the results of regression tests by performing comparisons between databases after the execution of successive test runs.

For example, an existing procedure may be:-

- Backup test database.
- Execute test runs with unconverted applications.
- Unload the data.
- Restore test database and re-run tests with converted applications.
- Unload the data.
- Compare the unloaded files using utilities such as FILE-AID or other compare utilities.

EZ-Extract Utility simplifies this procedure by allowing comparison between the updates written to the EZ-Image VSAM files. By executing the tests against image files, the user has all of the benefits of EZ-Image in respect of the DBA procedure for backing up and restoring data. In addition, as the images contain the updates only, the actual file comparison will be much quicker than a compare of the entire database. EZ-Extract will de-block each page in the image file to its CA-IDMS record components. Any compressed records are decompressed.

The procedure for a regression test using *EZ-Extract* then would be as follows:-

- Execute test runs with unconverted application writing updates to image.
- Run *EZ-Extract* against image creating sequential extract file
- Initialise image, and run test using converted application.
- Run *EZ-Extract* against image creating second sequential file.
- File compare extracted sequential files.

The image extract is performed by the EZ-Extract utility (DBIMEXTR) which is described in the next section.

7.2 Executing DBIMEXTR

DBIMEXTR is a utility that can be used to read all, or a subset, of database pages from a EZ-Image VSAM file and output each record on each database page to a sequential file in the format described by the dsect #DEXDS.

Data on the database page may be compressed. DBIMEXTR issues a bind run unit using the nominated subschema to obtain information relating to database procedures and will decompress data before writing it out to the sequential extract file.

The data written out comprises a record header, described by #DEXDS, followed by the record prefix then the record data. Either or both of the record prefix and data may be compared from the output of other runs to identify differences in set positions or data.

An end of job report is written to SYSLST listing all record types output with a count of the number of records extracted for each record type. It will also list all record ids located in the selected page group or area(s) that are not contained within the subschema.

The following figure (**Figure 19**) shows the JCL required to execute the DBIMEXTR utility.

```
//jobcard
//*
//DBIMEXTR EXEC PGM=DBIMEXTR,REGION=4096K
//STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//         DD DSN=CAIDMS.DBALOAD,DISP=SHR
//         DD DSN=CAIDMS.LOADLIB,DISP=SHR
//DICTDB   DD DSN=CAIDMS.DDLDML,DISP=SHR
//DCMSG    DD DSN=CAIDMS.DDLDCMSG,DISP=SHR
//DCLOD    DD DSN=CAIDMS.DDLDCLOD,DISP=SHR
//SYSUT1   DD DSN=dbimage-vsam-file,DISP=SHR
//SYSUT2   DD DSN=dbimage-extracted-data-file,DISP=(,CATLG),
//            DCB=(RECFM=VB,LRECL=halftrack-4,BLKSIZE=halftrack),
//            SPACE=(CYL,(20,100)),UNIT=SYSDA
//SYSLST   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIDMS  DD *
DMCL=dmclname
//SYSIPT   DD *
SUBSCHEMA=subschema-name
DBNAME=dbname
PG=page-group
AREA=area-name
DATE=ONLY
//*
```

Figure 19 DBIMEXTR Run-time JCL.

The JCL should be modified as follows:-

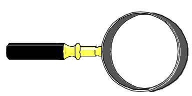
EZ-DB.CDMSLIB	The load library containing the EZ-DB run-time modules.
CAIDMS.DBALOAD	The library containing the DMCL and subschema load-modules
CAIDMS.LOADLIB	The load library containing the CA-IDMS run-time modules.
CAIDMS.DDLDML	The CA-IDMS dictionary DML area.
CAIDMS.DDLDCMSG	The CA-IDMS dictionary message area.
CAIDMS.DDLDCLOD	The CA-IDMS dictionary load area.
dbimage-vsam-file	SYSUT1 specifies the name of the EZ-Image VSAM file containing the updated pages from which you wish to perform the extract.
dbimage-extracted-data-file	SYSUT2 specifies the name of the sequential file to which the extracted data will be written. Note that this file should be allocated as follows:- `DCB=(RECFM=VB,LRECL=halftrack-4,BLKSIZE=halftrack)`

The following describes the run-time SYSIDMS parameter(s) which may be specified.

DMCL=*dmcl-name*	Name of DMCL

The following describes the run-time SYSIPT parameter(s) which may be specified.

SUBSCHEMA=*subschema-name*	This is a mandatory parameter. The subschema must contain all records required to be extracted. Note that if you wish to limit the extract to specific record types then provide a subschema with just the record types required.
DBNAME=*dbname*	Optional parameter if required for the bind run unit. This may be a dbname valid for either the base CA-IDMS database or the EZ-Image database.
PG=*nnn*	Optional parameter to limit the extract to the specified page group. Only one page group may be specified.
AREA=*area-name*	Optional parameter to limit the extract to the specified area name(s). This parameter may be repeated as often as required.
DATA=ONLY	Optional parameter forces EZ-Extract to extract the data portion of records only without the record prefix.



The RDW in DEXRLEN becomes the new length of the record output. DEXLILEN is adjusted to the length of the data portion of the record and DEXLIPL is set to zero.

7.3 EZ-Extract and EZ-Bridge

There is the potential for misleading *miss-matches* to occur when File-Comparing extracted Image files. These can be the result of database restructures, for example to expand a date field. Consider the following scenario. A user runs *test case-1*, using the old application and database structure. The image data is extracted. The user restructures the underlying database to expand a date field, and then runs *test case-2* using the converted application. Because of the restructure and record size changes, it is possible that records will be relocated and database page differences may appear even in un-related data. This may reduce the usefulness of the File Compare results.

These problems can be avoided by using the **EZBridge** product. **EZBridge** is a component of EZ-DB which assists users in the testing and roll-out of applications by allowing the implementation of database *restructures* and the necessary program *re-compilations* to be performed independently. *Either* the recompilation *Or* the restructure can be performed first, with **EZBridge** ensuring that the database record is converted to the *old* or *new* format as necessary.

For example, referring to the previous scenario. The user could run *test case-1* as above using the *old* application and database structure. After performing the *image extract*, using EZBridge the user could run *test case-2* using the converted application without having to restructure the database. After performing the second *image extract*, a File Compare would only detect genuine miss-matches of data.

Alternatively, the user could restructure the database before the first test case, and using EZBridge, execute both test cases against the restructured database.

7.4 EZ-Extract Data File DSECT (DEX)

The following documents the DSECT which is used to describe the records written by DBIMEXTR to the extracted file. This information may be required depending on the type of file compare you are performing.

```
***
***
***  DEX:      EZ-IMAGE EXTRACT DATA FILE DSECT
***
***
DEX      DSECT                              02/13/90 15:08:26   03/11/91
DEXRLEN  DS     H                        RECORD LENGTH
         DS     H                        ZEROES
DEXRECNM DS     CL18                     RECORD NAME
DEXPGRUP DS     H                        PAGE GROUP
         DS     F                        SPARE
DEXDBKEY DS     F                        DATABASE KEY
DEXRECID DS     H                        RECORD I.D.
         DS     H                        SPARE
DEXLILEN DS     H                        DATABASE RECORD IMAGE LENGTH (LI)
DEXLIPL  DS     H                        PREFIX LENGTH (LI)
         DS     F                        SPARE
DEXDATA  DS     0C                       START OF RECORD PREFIX PLUS DATA
DEXDSLEN EQU    *-DEX                    DSECT LENGTH
```

Technical Notes

8 Technical Notes

8.1 EZ-Image Performance Tuning

This section gives some hints and tips on tuning and setting up your EZ-Image environment.

If you are using the EZ-Image databases in a high throughput environment, it is important to minimise the VSAM IO.

There are a number of parameters available when you setup your EZ-Image environment that affect the performance of your EZ-Image databases:

8.1.1 DATABUF and INDXBUF

These options are defined on the **DMCL** entry of your **COIMAGE** table. They are VSAM buffers that are used for moving data between virtual and auxillary storage.

DATABUF specifies the number of buffers to made available for transferring the data portion of the VSAM file. The default is 100, but by increasing this value you can improve the chance of EZ-Image finding the required page already in the VSAM buffer, thereby reducing the physical IO required.

INDXBUF specifies the number of buffers to made available for transferring the index portion of the VSAM file. The default is 150, but in systems where you expect a high number of IOs against the EZ-Image VSAM files, you can improve performance by increasing the value of INDXBUF. Ideally the entire index portion of the VSAM file should be held in the buffers, to maximise the throughput to the VSAM file. You can use the MVS command **LISTCAT** to obtain details about the number of records the index contains and then set INDXBUF accordingly. If you use this method to set INDXBUF, you should periodically rerun LISTCAT to check on the size of the index, and reset INDXBUF.

8.1.2 Multiple DDNAMEs

This is the parameter on the **PG** entries of your **COIMAGE** table. It specifies which EZ-Image VSAM files the EZ-Image Page Groups will use.

Up to eight VSAM files can be defined to a single EZ-Image Page Group. Under normal circumstances most users will probably allocate one image page group to one VSAM file. In high throughput systems with a large number of IDMS files being imaged, it will be easy to become bottlenecked on the VSAM.file

For example, if you are imaging a page group that covers 100 CA-IDMS physical files, then if you are running a high throughput system against the image page group, it is easy to see how the system will bottleneck as 100 files IOs are being converted to IOs against the single image VSAM file.

It is therefore recommended that in high throughput EZ-Image systems, you make full use of the multiple VSAM file processing to spread the IO load across multiple files.

For example, suppose you wish to image page group zero on your production database. Where the entire page range of the database is from 0 to 120,000, and the dictionary occupies the range 0 to 20,000, which you do not want imaged.

The following sample 'COIMAGE ENTRY=PG' statement will create an image page group 50, which will ignore the dictionary areas and then split the imaged records across four VSAM files;

In this example **LOPAGE=20001** causes EZ-Image to only start imaging page group 0 from page number

```
       #COIMAGE ENTRY=PG,                                           X
              IDMSPG=0,                                             X
              IMAGEPG=50,                                           X
              LOPAGE=20001,                                         X
              DDNAME=(COIMVS1,50000,COIMVS2,70000,COIMVS3,90000,    X
              COIMVS4,120000),                                      X
              .
              .
              .
```

Figure 20 Sample #COIMAGE PG entry using multiple DD names.

20,001 - ie ignoring the dictionary areas. VSAM file DDname COIMVS1 will be used to image pages in the range 20,001-50,000, COIMVS2 will be used for pages 50,001-70,000, COIMVS3 will be used for pages 70,001-90,000 and COIMVS4 for pages 90,001-120,000.

You can specify EZ-Image to use a single VSAM file for multiple EZ-Image Page Groups. This is however only recommended on systems you know will have a very low throughput.

8.2 Region Size and Warmstart

It is possible for CA-IDMS to get GETMAIN failures allocating its XA storage pools after a Warmstart with EZ-Image active. This may occur if the total available XA storage is insufficient for both the CA-IDMS XA pools and the EZ-Image VSAM I/O buffers, whose size is determined by the DATABUF and INDXBUF parameters in the COIMAGE table.

At warmstart, EZ-Image will allocate the VSAM buffers *before* CA-IDMS GETMAINs its XA pools. If there is then insufficient XA storage for all of the CA-IDMS XA pools, the CA-IDMS GETMAIN will fail and the CV Abends.

If there is no warmstart, then CA-IDMS allocates its pools first without a problem. If there is then insufficient XA storage for the EZ-Image VSAM I/O buffers but there *is* sufficient storage available in 24-bit storage, the buffers are allocated *below-the-line* and EZ-Image is able to start up.

It is recommended therefore that a warmstart test should always be performed after installing EZ-Image or changing the size of EZ-Image or CA-IDMS pools. If it is necessary to make more XA storage available then increase the region size as required.

Refer to Technical Note **CD602.T02** which contains additional information about region size and in particular the IEFUSI exit.

If it is necessary to decrease the size of the EZ-Image I/O buffers then specify smaller values through the DATABUF and INDXBUF parameters in the COIMAGE table.

8.3 Initialising VSAM files and Flushing Buffers

This Technical Note was previously published as tech note **CG412.T11**

Users of DB-Image, who initialise the DB-Image VSAM files while CV is still active may, after successful initialisation, still find they pick up copies of the old image pages. This is due to the image pages still being in the CA-IDMS buffers. To prevent unwanted image records being retrieved, users should ensure that the CA-IDMS buffers are purged of the image pages. This can be done in one of two ways;

1. Use DCMT to vary the DB-Image database area OFFLINE then ONLINE. This will flush any pages from the buffers.

2. Alternatively you can issue a DCMT VARY AREA PURGE command for each DB-Image area. This will temporarily stop IO for the area, flush any pages from the database buffers, then resume IO.

It is recommended that the DB-Image areas are varied offline before the VSAM file initialisation is performed, then after successful initialisation vary the areas back online. This will prevent any IO being attempted while the VSAM file is being initialised, as well as flushing the buffers.

8.4 Restructure and Reorganisation of Image databases (REORG=YES|NO)

This Technical note is based upon the previously published tech note **CD602.T20**

There are special considerations which apply when you wish to reorganise an underlying database against which you have implemented one or more *images*. Typically, the image VSAM files will only contain a small proportion (e.g. 20%) of the pages in the underlying database, i.e., those that have been updated. If you were to reorganise the database (e.g., using Unload/Reload or Restructure), this would result in most or all of the pages being updated, and therefore being written to the VSAM file, with a consequent higher DASD requirement.

To circumvent this problem, you should specify the parameter **REORG=YES** when running the database reorganisation. This parameter can be specified in either the DBIMCTL or COXACTL parameter file.

When REORG=YES is specified, only pages found in the image file are updated and written back to the image file. <u>No updates are written to the underlying database</u>.

The procedure for reorganising/restructuring a base database and it's associated *images* should be as follows:-

(i) Restructure/Reorg EZ-Image database specifying REORG=YES
(ii) Repeat (i) for each EZ-Image database.
(iii) Restructure the base CA-IDMS database.

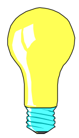
Now the base database and all images are restructured.

Note that this process will only work if the database being updated by the restructure or unload/reload is written back to the same database page that it originated from. i.e. The restructure does not result in fragments being created and there have been no CALC or VIA overflows that would result in an unload/reload loading data into a page other than from which it originated. It is recommended therefore that you provide for sufficient freespace, for example by using XPAG prior to the restructure.

8.5 Running Utilities and local mode batch jobs (IDMSUTIL=YES|NO)

This Technical note is based upon the previously published tech note **CD602.T20**

It is common to run local mode batch jobs (including CA-IDMS utilities) with a local mode DMCL containing a single database definition and to change the run-time JCL to point the DDNames to the database files to be processed by that run.



For example, you have ten identical test databases defined to your CV with page groups 51-60. You have a local mode DMCL with a single database definition with page group zero. When you run IDMSDBAN, you submit the job once for each test database using the same DDNames each time, but changing the DSNames to point to the appropriate database.

In this case, you should specify the DBIMCTL run-time parameter **IDMSUTIL=YES** in the local mode JCL.

Specifying IDMSUTIL=YES instructs EZ-Image to use the database definition in the DMCL and not to construct new page group(s) and/or segment(s). EZ-Image will still process any data for that database held in its image VSAM file normally.

The applicable *image* database is identified by including the appropriate COIMAGE table or DBIMCTL parameters.

Another use for this parameter is in CA-IDMS R12.1 and R14 when running certain BCF utilities, (e.g.,. IDMSDBAN and PRINTSPACE) against *image* databases.

In this case, it is necessary to run the utility against the *base* database specifying **IDMSUTIL=YES**. ie., use DBNAME for the base, but use the appropriate COIMAGE table or DBIMCTL parameters to identify the applicable *image*. Additionally, you may cause the utility to write back to the image VSAM files <u>only those pages that were originally read from the VSAM file</u> (and no updates to the underlying database) through the **REORG=YES** parameter (as described in the previous section).

8.6 Using EXIT23 and DBNODE to set the DBNAME

The method of selecting the applicable *image* database for a particular online user or batch job is through the DBNAME mechanism. However, this mechanism is problematic in particular circumstances.

One such example which has come to light, is where databases are *segmented*, for example by geographic region or account range. In this scenario, the user would typically have a high-level interface or common module which would determine which segment should be used and then issue a BIND DBNAME=*segment-name* as applicable. This would obviously conflict with any requirement to point the application at an *image* database. i.e., The DBNAME=*segment-name* overrides any other DBNAME requested eg., through DCUF SET DBNAME.

To resolve this issue, it is now possible to use the combination of the DBNODE parameter and a special DC EXIT23 (pre-bind), as a means of overriding the DBNAME used at BIND time. The user can then enter DCUF SET DBNODE=xxx either under CICS or IDMS. The pre-bind exit locates the DBNODE and uses it to override the DBNAME set by the application.

For example, you could set DBNODE=IMAG*nn*; IMAG is recognised as a keyword to trigger the DBNAME override, and *nn* is then suffixed to the applications DBNAME to generate the *image* segment name. Eg., CUSTA becomes CUSTA*nn*.

Note that batch jobs may also have DBNODE specified via a SYSIDMS parameter.

Another situation where this mechanism may be required is in the case of CICS users. A CICS transaction DCUF is provided which permits CICS users to set the DBNAME for native CICS-DML programs. The CICS **DCUF SET DBNODE** transaction is used when executing CICS/UCF intermittent applications, allowing the DCEXIT23 to set the DBNAME when the bind is performed on the back-end. Further details of implementing and using the CICS DCUF transaction are provided in **section 8.7**.

The supplied source (DCEXIT23) may be tailored to suit the particular sites requirements. If used, DCEXIT23 must be added to the RHDCUXIT module. Refer to the CA-IDMS *System Operations* manual for further details of installing DC/UCF exits.

8.7 Installing and using the CICS DCUF transaction.

To enable a CICS user to set a CA-IDMS DBNAME for a particular terminal without application amendments, the CICS transaction **DCUF** is provided. Using DCUF, the DBNAME can be set for native CICS-IDMS programs (i.e., CICS programs issuing CA-IDMS DML) and CICS/UCF intermittent applications.

Note that implementation of the DCUF facility requires modifications to the IDMSINTC and/or IDMSINTL interface. The full description of how to implement the DCUF facility, including the required source modifications to the CA-IDMS CICS interface, is provided in PTF **CD710.002** which can be found in the installation tape PTF library.

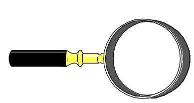
Once implemented (as described above) the syntax of the DCUF transaction is as follows:-

DCUF <SET> DBNAME=*dbname* **DBNODE=***dbnode*

Note: The **SET** is optional. Either or both of DBNAME and DBNODE may be entered.

DBNAME

The CA-IDMS DBNAME is set for a particular terminal to *dbname,* where *dbname* is 1-8 alphanumeric characters which should correspond to a valid DBNAME used by the applicable CA-IDMS CV.

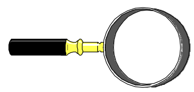


This DBNAME is applicable to native CICS applications only. The DBNAME specified will be moved to Subschema control before any subsequent Bind Rununit.

All tasks executed from this terminal will use the DBNAME specified until varied by another DCUF transaction. Enter the DCUF transaction with no DBNAME and the DBNAME will no longer be used.

DBNODE

The CA-IDMS DBNAME is set for a particular terminal to *dbnode,* where *dbnode* is 1-8 alphanumeric characters which should correspond to a valid DBNAME used by the applicable CA-IDMS CV.



This DBNAME is applicable to CICS/UCF applications only. The DBNAME specified via the DBNODE parameter will be used by the DCEXIT23 (*pre-bind*) exit when the Bind Rununit is executed on the DC back-end. Refer to section 8.6 for further details of the *pre-bind exit.*

All tasks executed from this terminal will use the DBNODE specified until varied by another DCUF transaction. Enter the DCUF transaction with no DBNODE and the DBNODE will no longer be used.

8.8 Imaging native VSAM files

1. Native VSAM files may be imaged, but the image files may be in retrieval only. If segments that include native VSAM files need to be imaged, and the native VSAM files in the image segments need to be updated, then you will have to:-

   (a) Define the image segments containing native VSAM to CA-IDMS, using the new page group, the same as you would if setting up a new test database without using EZ-Image.

   (b) Create physical copies of the native VSAM files for each new page group. CA-IDMS files in the image segment do not need to be duplicated, only the native VSAM files.

   (c) In the COIMAGE parameters, include the new Image segment but with IDMSSEG and IMAGSEG both equal to the Image segment name, and IDMSPG and IMAGEPG the image page group.

2. Any native VSAM files being imaged must be included in the IDMS start up JCL. eg., if the base DDNAME is VSAMFIL1 and the Image DDNAME is VS99FIL1 then add to the IDMS start up JCL file:

```
//VS99FIL1 DD DSN=dsn,DISP=SHR
```

8.9 *Hot Backup* questions and answers

Q. When I use the command DBHOT STOP SEGMENT=seg-name, EZ-Image updates the physical database even if the segment is offline in CV. How does EZ-Image write to the database when it is offline?

A. *EZ-Image issues BDAM writes directly to the CA-IDMS file - these are totally independent of CA-IDMS.*

Q. How big should the VSAM-file be to handle the updates while the backup is taken? Is there any rule of thumb that says, for instance, how many updates are handled by one cylinder (3390)?

A. *There is no simple rule of thumb, because it is going to depend on both your CA-IDMS page sizes, and by how much EZ-Image can compress the pages on the VSAM file. You should "err" on the generous side. We would suggest as a starting point - estimate the <u>number of pages</u> being updated during the hot backup period, multiply these by the <u>largest page size</u>, this will automatically give you a safety margin due to using the large page size, and no allowance for compression. But then add a large* *percentage to this (at least 100%) to be sure of not running out of space. This is to allow extra space should your hot backup window be extended for unexpected reasons. One large user of Hot Backup does a similar calculation to the above on a regular basis and then allocates the VSAM file at SIX times the size! This then gives them plenty of time should they have production problems while running Hot Backup without having to worry about the VSAM file getting filled.*

Q. What happens if the VSAM-space goes full while we are running the hot backup? Will the updates wait or will the tasks ABEND?

A. *Tasks will ABEND with IO errors and the run-unit will be suspended as they will be unable to rollback due to the rollback still trying to write to the VSAM file. Therefore always "err" on the side of extra large VSAM files for the Hot Backup Image file. There is a tech note (CG602T18) that explains how you can remove these suspended run-units, but you should aim to never be in such a position!*

Q. When we are finish with the hot backup and use the DBHOT STOP command to update the database, what happens with new updates from the users while EZ-IMAGE is writing down the pages to the database? Will they continue to write to the VSAM-dataset, will they wait, or will they be written directly to the database at the same time EZ-IMAGE is writing down the updated pages?

A. *Pages are written back in page order, therefore each CA-IDMS file is processed in order. As each CA-IDMS file is being processed by the DBHOT STOP program, updates to that particular file will wait. Once a file has been processed by DBHOT STOP it is then given back to CA-IDMS to carry on updating as normal, while subsequent files get processed by DBHOT STOP. Updates to files that DBHOT STOP has not yet reached in its processing, will continue to be written to the Image VSAM file.*

Notes