Syncsort Optimize IDMS

(EZ-IDMS)

Syncsort IDMS Reorg/RSTU User Guide

Release 6.0

Version 6.10

March 2021

Syncsort Optimize IDMS Reorg/RSTU User Guide
(formerly EZ-Reorg/RSTU User Guide)

NOTICE

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.

PREFACE:

Syncsort IDMS Reorg (formerly EZ-Reorg) is a component of Syncsort Optimize IDMS (formerly EZ-IDMS or EZ-DB) which allows users of CA IDMS to perform database *reorganizations* while the affected area(s) are still available for update.

Syncsort IDMS RSTU (formerly EZ-RSTU) is a component of Syncsort Optimize IDMS (formerly EZ-IDMS or EZ-DB) which allows users of CA IDMS to perform database *restructures* while the affected area(s) are still available for update.

TABLE OF CONTENTS

FIGURES

New Features

The following changes and enhancements have been implemented in Release 6.0 (CL1001) for CA IDMS releases 18.0/18.5/19.0.

MIXED/MULTIPLE PAGE GROUP SUPPORT

CA IDMS 18.5 introduced Reorgs by DBNAME with index sets spanning segments. R6.0 of Syncsort IDMS Reorg implements DBNAME and cross segment index set support.

This has necessitated some noticeable changes to the Syncsort IDMS Reorg procedures and changes to various Syncsort IDMS Reorg file structures. If you are an existing user of Syncsort IDMS Reorg/RSTU and have saved tailored JCL jobs for Syncsort IDMS Reorg/RSTU, then these must be updated/replaced using the sample jobs provided in the R6.0 install SRCLIB.

The following changes and enhancements have been implemented in Release 5.5 (C40901) for CA-IDMS release 16+ only.

STAMP=YES

STAMP=YES in now mandatory. Also, if updates are found before the stamp then it is no longer possible to restart DBEZRFWD. If you have been using STAMP=NO you must change your procedures to include the running of the DC/UCF task DBEZSTRT when the areas involved in the SYNCSORT IDMS REORG or Syncsort IDMS RSTU are quiesced.

This is to ensure that the first journal processed by SYNCSORT IDMS REORG or Syncsort IDMS RSTU is the correct journal.

Refer to section 3.4.2 for further information about DBEZSTRT.

File Allocations

Users upgrading from R5.3 should be aware that some records lengths have changed for this release.

Users who wish to continue using existing JCL that has been set up, should review the record lengths for DBK2/DBK3/DBK4/SR23/RUPD/RUPS/CFRU and CNTL files in INSTDBEZ and DBEZJOB0 members, and change their JCL accordingly

Refer to Section 4.4 for further details about file allocations.

The following changes and enhancements have been implemented in Release 5.4 (C10040) for CA-IDMS Release 15.0+ only.

DATA SHARING

There is a new separately chargeable Data Sharing option which allows merged journal input.

The following changes and enhancements have been implemented in Release 5.3 (C90701) for CA-IDMS Release 14.1 and earlier.

STAMP=YES

There is a new parameter STAMP=YES/NO (default Yes). This is to allow the check in DBEZRFWD for the Syncsort IDMS Reorg Journal stamp to be bypassed.

DBEZDB2K=VSAM

Specify the optional parameter DBEZDBK2=VSAM if you wish to use a VSAM file for DBEZDBK2. The advantage of a VSAM file is that DBEZSXT3 and DBEZSXT4 will run much quicker and DBEZCOPY is no longer executed.

New DBEZJOB4 job

DBEZJOB4 is run in place of DBEZJOB3 to create the old/new DBKey file (DBEZDBK2) if using a VSAM dataset for this file (see DBEZDBK2=VSAM parameter).

Introduction

Syncsort IDMS Reorg and **Syncsort IDMS RSTU** provide functionality that enhances the operation of the standard **CA IDMS Unload/Reload** and **Restructure** processes.

Syncsort IDMS Reorg and **Syncsort IDMS RSTU** are used **in addition** to (not in place of) the standard CA IDMS utilities which provide the basic maintenance functionality (i.e, UNLOAD/RELOAD utility (BCF UNLOAD in release 12.0+) or DB-REORG for database reorganization, or IDMSRSTU (BCF RESTRUCTURE in release 12.0+) for database restructure. It is therefore assumed throughout this User Guide that the reader is familiar with the operation of those CA products.

It should be noted that **Syncsort IDMS Reorg** and **Syncsort IDMS RSTU** are separately licensed components of the **Syncsort Optimize IDMS** product set (formerly EZ-DB or EZ-IDMS). However, because there is an overlap in the functionality and operation of the two products when both are implemented, for clarity they are both described in this User Guide. The reader should be aware of which product(s) are implemented, and keep in mind that some sections of this User Guide may not be applicable.

For sites using CA IDMS DATA SHARING, there is a separately licensed option to allow MERGED JOURNAL input.

This introduction describes the contents of the sections included in this document.

- **Section 1 - Syncsort IDMS Reorg/RSTU Overview**.

  This section introduces the basic concepts of Syncsort IDMS Reorg® and Syncsort IDMS RSTU®

- **Section 2 - Syncsort IDMS Reorg/RSTU User procedures Overview**

  This section provides an overview of the different phases of the Syncsort IDMS Reorg/RSTU process.

- **Section 3 - Syncsort IDMS Reorg/RSTU User Procedures Detail.**

  This section describes each phase of the Syncsort IDMS Reorg/RSTU process in detail.

- **Section 4 - Technical Notes**

  This section provides more technical information where applicable.

- **Section 5 – Appendix**

  This section includes DSECTS and Operating System dependent JCL.

Syncsort IDMS Reorg/RSTU Overview

1 Syncsort IDMS Reorg/RSTU Overview

1.1 24x7 Availability

Many sites these days strive towards non-stop systems availability (also referred to as 24x7). Quite often, business circumstances dictate that loss of availability be minimal or non-existent. Many users can quantify in financial terms what each hour or minute of down time costs their business in lost revenue or lost opportunity.

Through its evolution **CA IDMS** has gradually introduced many 24x7 features. Release 12.0 addressed many of the important aspects of 24x7 operation (for example dynamic file allocation and dynamic DMCL modification), and more 24x7 features have been implemented in subsequent releases.

One area not addressed by CA, but critical to many users, is that of database maintenance such as **database reorganization** and **restructure.** For sites requiring 24x7 availability, the required down-time to perform an **Unload/Reload** in particular is a major problem. Depending on the sizes of the affected areas, it is not uncommon for "*reorg's*" to take a whole week-end to complete. Often, they are planned months in advance, for instance to coincide with long-weekends. They can be expensive in terms of system down-time and staff overtime. Because of the time criticality, there is no margin for error. If things go wrong, it may be necessary to give up, meaning that many months may go by before there is another opportunity to implement crucial maintenance.

Similarly for database restructure; Although a database restructure (typically) does not take as long as a database reorg, it is still a requirement that the database be offline while the restructure takes place. With the number of restructures likely to increase, implementation of database restructures is likely to become increasingly problematic for 24x7 users.

Even for non-24x7 shops, the ability to schedule reorg's and restructures during normal working hours (with the affected areas still fully available to end users) gives CA IDMS users an unprecedented degree of flexibility in terms of scheduling and implementation of routine database maintenance.

1.2 Syncsort IDMS Reorg/RSTU Operation

With Syncsort IDMS Reorg/RSTU, these problems become a thing of the past. Syncsort IDMS Reorg/RSTU allow CA IDMS users to perform database reorganizations and restructures **while the affected areas are still available for update by the production system**.

In simple terms, this is achieved as follows:

- Take an in-flight backup of the affected area(s) - production remains up
- Perform restructure and/or reorganisation against copy - production remains up
- Feed the off-loaded journals into **Syncsort IDMS Reorg/RSTU** “catch-up” process. This process analyses the off-loaded journals and issues the required DML against the restructured/reorganised database, until all off-loaded journals are processed - production remains up.
- Shutdown production. Off-load the “active” journal and feed into **Syncsort IDMS Reorg/RSTU** “catch-up” process - production down.
- Optionally, take back-up- production down.
- Start-up system using newly restructured/reorganised area(s).

A comparison of the implementation of a database reorganization with and without Syncsort IDMS Reorg/RSTU is shown in the following figure (Figure 1).

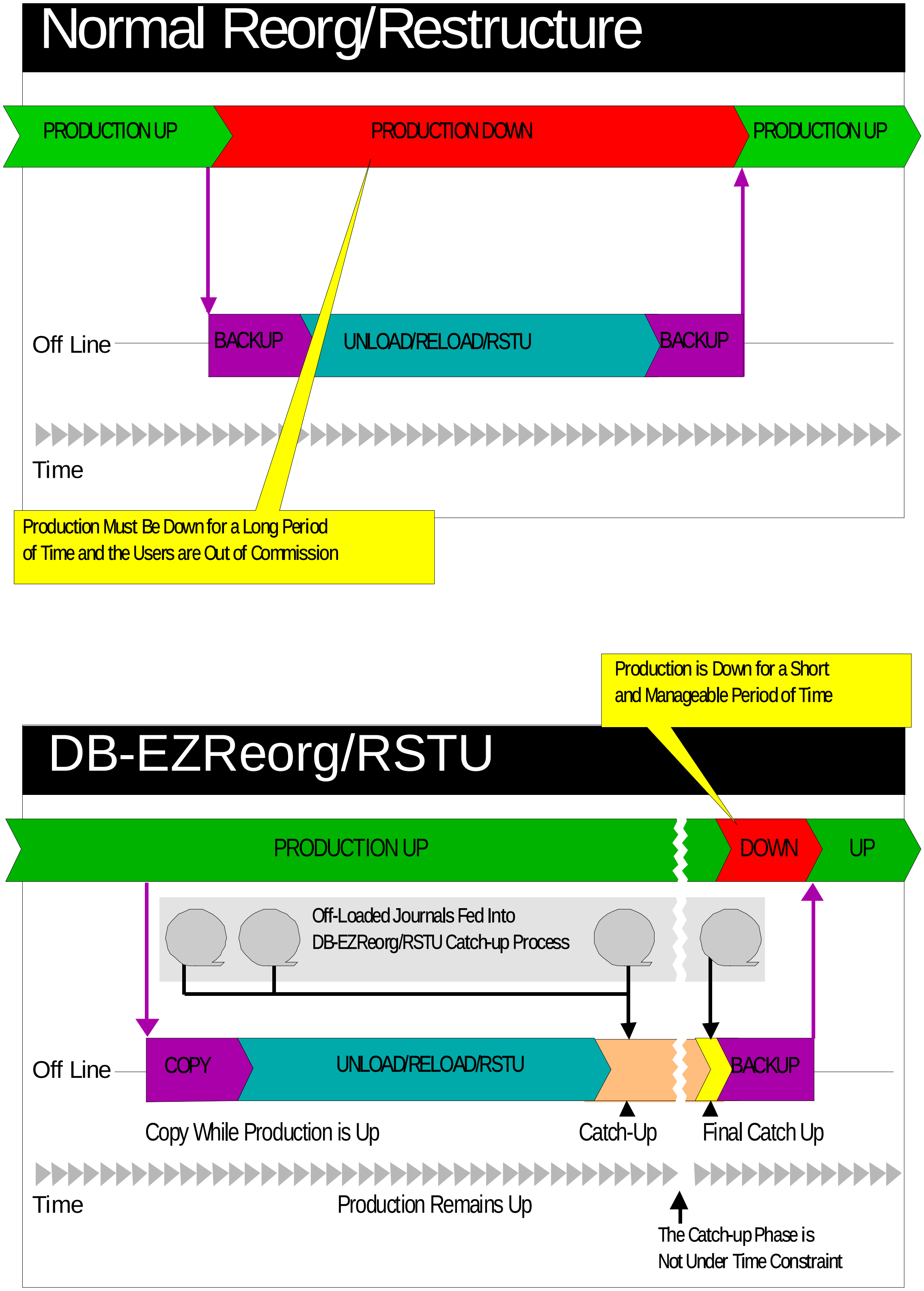


Figure 1 Syncsort IDMS Reorg/Syncsort IDMS RSTU Overview

1.3 Scheduling Considerations when using Syncsort IDMS Reorg/RSTU

Because of the requirement for production systems to be shut-down (or affected areas to be in retrieval mode) while a reorganisation or restructure took place, reorg's and restructures have typically, in the past, been scheduled for overnight or weekend implementation. Also, because of the requirement for systems to be unavailable while the reorg/restructure took place, it was often the temptation to try to do as much as possible in “one fell swoop”. That is, combining many, often un-related, areas in one implementation. This in itself caused prolongation of the run-time. The philosophy being, “let’s get all of this over in one go”!

With **Syncsort IDMS Reorg/RSTU,** you can re-think this approach. In theory, you no longer need be concerned with the length of time required to perform the Reorg/Restructure or the initial “catch-up” phase, as the production system is fully available throughout this period. The duration of the “catch-up” phase will be dependent on the amount of update activity which has occurred against the affected areas since the start of the initial in-flight backup. This will obviously affect the number of journal records which will have to be processed by “catch-up”. Similarly, the amount of update activity will affect the duration of the final “catch-up” which takes place with the system unavailable.

It would therefore be recommended that users might still wish to restrict the update activity which would occur while this process was being carried out. For example, it would not be advisable to run large batch updates against the affected areas. You may even choose to limit online activity to crucial transactions only. However, it is not suggested that this is a requirement. As the main “catch-up” phase takes place with the production areas fully available, the “catch-up” phase itself is not under any time constraint.

1.4 Syncsort IDMS Reorg/Syncsort IDMS RSTU Features Summary

- Reorganize CA-IDMS SQL and non-SQL databases while available to end-users for retrieval and update. All functions of Unload/Reload are fully supported. No restrictions apply. (Syncsort IDMS Reorg)

- Restructure CA-IDMS SQL and non-SQL databases while available to end-users for retrieval and update. All functions of Restructure are fully supported. No restrictions apply (Syncsort IDMS RSTU).

- Supports CA IDMS release 17.0, 18.0, 18.5 and 19.0

- Index Rebuilds fully supported (Syncsort IDMS Reorg).

- SQL databases fully supported (Release 12.x and above).

- UNLOAD/RELOAD utility and DB-Reorg product fully supported.

- Operates in the IBM z/OS environment.

- Supports mixed page groups and cross segment index sets (DB-Reorg under Release 18.5).

- Supports Merged Journal input from a Data Sharing environment.

User Procedures Overview

2 User Procedures Overview

The Syncsort IDMS Reorg and Syncsort IDMS RSTU products consist of a number of programs and procedures which must be executed, before, during and after the Restructure and/or Unload/Reload process. The actual Restructure and/or Unload/Reload itself is carried out according to your current procedures, for example, using IDMSRSTU utility, the DB-Reorg licensed product, the standard BCF Restructure/Unload/Reload.

The Syncsort IDMS Reorg/RSTU installation source library contains skeleton JCL required for execution of the Syncsort IDMS Reorg/RSTU job-steps. These job steps can be found for each applicable operating system in the Appendix of this user guide. This user guide describes the job-steps, when they should be run, and the other required procedures and when they should be performed.

There are certain steps and procedures which may overlap. Where this is the case, it is explained in the User Procedure Detail section later. For simplicity, however, you may choose to perform the procedures and execute Syncsort IDMS Reorg/RSTU programs one after the other until you are more familiar with the product.

2.1 Summary of Job Steps and User Procedures

The entire Syncsort IDMS Reorg/RSTU process consists of the following Syncsort IDMS Reorg/RSTU job-steps and user procedures.

Syncsort IDMS Reorg/RSTU job step or user Procedure	Description	When to Run
INSTDBEZ	Allocates all work files.	Any time prior to commencement of rest of process.
DBEZJOB0	Initialises DBZSR23, DBZDBK2, RECPGXRF and the “carried-forward files” (DBZCFRU DBZCFJN) which must be empty prior to the 1st iteration of DBEZRFWD.	Any time prior to commencement of rest of process.

Syncsort IDMS Reorg/RSTU job step or user Procedure	Description	When to Run
DBEZJOB1	Initializes DBZCNTL (control file) which controls checkpoint/restart process, also defines scope of run, eg. areas being reorganised or restructured. Identifies dependent areas and records stored direct. Generates the RECPGXRF cross reference file containing old/new page group details for all records in the subschema.	Any time prior to commencement of rest of process.
Create database for restructure/unload	You must take a copy of the database areas to be restructured or reorganised. This may be a static backup (ie. taken while area(s) are in retrieval or offline) or a hot-backup depending on your current procedure.	When you are ready to commence database restructure or reorganisation process.
DBEZJOB2	Runs against copy of areas to be processed. Creates file identifying SR2/SR3 records to be used by DBEZRFWD (“Catch up”) process. Also, creates file identifying logically deleted records which may be used as input to IDMSLDEL process. (Saves running IDMSDUMP reports).	Can be run as soon as you have created the copy of the areas to be processed.
Perform database restructure	Restructure the database according to your current procedure. For example using the BCF. Note: The restructure is performed against the copy created previously.	Soon after you have completed DBEZJOB2.
Perform database reorganisation	Reorganise the database according to your current procedure. For example, using the DB-Reorg (CA licensed product) or standard unload/reload utilities. Note: The reorganisation is performed against the copy created previously.	Soon after completion of DBEZJOB2 if you are using UNLOAD/RELOAD or reorg Soon after you have completed the restructure if you are combining restructure and reorg in a single “catch-up”

Syncsort IDMS Reorg/RSTU job step or user Procedure	Description	When to Run
DBEZJOB3 Contains details of the alternative jobs, dependent on whether you are using UNLOAD/RELOAD or DB-Reorg with a non-VSAM DBK2 file. **NOTE**: Not required if PROCESS=INDEX	**DBEZJB3R** is for DB-Reorg and non-VSAM DBK2 file: Runs against DBKEYS files produced by DB-Reorg to create an Old/New non-VSAM DBKey file and SR2/3 file required by DBEZRFWD ("Catch up") process.	After completion of DB-Reorg processing.
	DBEZJB3U is for UNLOAD/ RELOAD and non-VSAM DBK2 file: Runs against SYS005 file produced by IDMSDBL2 to create an Old/New non-VSAM DBKey file required by DBEZRFWD ("Catch up") process.	Usually, after completion of Unload/Reload process, prior to DBEZRFWD. In fact, can be run as soon as IDMSDBL2 has completed in parallel with final sorts if desired.
DBEZJOB4 is used instead of DBEZJOB3 if you have specified DBEZDBK2 is VSAM. Contains details of the alternative jobs, dependent on whether you are using UNLOAD/RELOAD or DB-Reorg with a VSAM DBK2 file. **NOTE**: Not required if PROCESS=INDEX	**DBEZJB4R** is for DB-Reorg and VSAM DBK2 file: Runs against DBKEYS files produced by DB-Reorg to create an Old/New VSAM DBKey file and SR2/3 file required by DBEZRFWD ("Catch up") process.	After completion of DB-Reorg processing.
	DBEZJB4U is for UNLOAD/ RELOAD and VSAM DBK2 file: Runs against SYS005 file produced by IDMSDBL2 to create an Old/New VSAM DBKey file required by DBEZRFWD ("Catch up") process.	Usually, after completion of Unload/Reload process, prior to DBEZRFWD. In fact, can be run as soon as IDMSDBL2 has completed in parallel with final sorts if desired.
DBEZRFWD	This is the Syncsort IDMS Reorg/RSTU "Catch-up" process. Input is offloaded journals from the production system and work files created by DBEZJOB2 and DBEZJOB3/4. Generates required DML to replicate any updates to affected areas against the restructured/reorganised copy.	As soon as restructure has completed, or in the case of reorg, as soon as unload/reload (and DBEZJOB2/3/4) has completed. This job is run multiple times, processing a "batch" of offloaded journals at a time, until ALL offloaded journals have been processed.

Syncsort IDMS Reorg/RSTU job step or user Procedure	Description	When to Run
Vary Affected Areas to Retrieval and Vary Journal	Final "Catch-up" has to be performed with affected area(s) in retrieval or offline. Vary Journal will cause the final active journal to be offloaded.	As soon as DBEZRFWD has completed processing all of previously offloaded journal files.
DBEZRFWD	Final "Catch-up".	As soon as active journal has been offloaded, "Catch-up" any previously offloaded journals not yet processed.
Implementation of restructured/reorganised database area(s)	Switch production system to use new area(s). May require system cycle or can be done dynamically.	After completion of DBEZRFWD final "catch-up". It is also recommended that you take a secure backup of old and new area(s) prior to implementation.

User Procedure Detail

3 User Procedure Detail

The following table identifies all of the Syncsort IDMS Reorg and Syncsort IDMS RSTU jobs and internal job-steps in addition to the user procedures with a cross-reference to the page in this user guide where the step is described in detail.

REORG	RSTU	JOBNAME	STEPNAME	Description		Page
✓	✓	DBEZJOB0		Run DBZINIT to allocate and initialise "carried forward" and work files.	3.2	20
✓	✓	DBEZJOB1	DBEZCNTL	Run DBZCNTL to initialise Control File	3.3	21
			SORTPGXR	Sort RECPGXRF recid/page group cross reference		
			SORTPGXR	Update RECPGXRF cross reference with new segment details		
✓	✓			Create database for Restructure/Unload	3.4	27
✓	✓	DBEZJOB2[1]	DBEZSR23	Identify SR2/SR3 Pairs	3.5	30
			SORTSR23	Sort SR2/SR3 DBKeys into ascending sequence.		
✗	✓			Perform Restructure using Standard procedures	3.6	34
✓	✗			Perform Unload/Reload using Standard Procedures	3.7	35
✓	✗	DBEZJB3R[2]	DBEZSXT6 DBEZSRT6	Generate File of OLD and NEW DBKeys from DB-Reorg output sorted on Page Group and DBKey	3.8.2	37
✗	✓	DBEZJB3U[2]	DBEZSXT7 DBEZSXT8	Generate File of OLD and NEW DBKeys from IDMSDBL2 output.	3.8.1	36
✓	✗	DBEZJB4R[3]	DBEZSXT6	Generate VSAM File of OLD and NEW DBKeys from DB- Reorg output	3.9.2	39
✗	✓	DBEZJB4U[3]	DBEZSXT7 DBEZSXT8	Generate VSAM File of OLD and NEW DBKeys from IDMSDBL2 output.	3.9.1	38
	✓	DBEZRFWD	DBEZCUP1	Extract Journal Data	3.10.3	47
			DBEZSUPD	Sort Run Unit Data	3.10.4	47
			DBEZSJNL	Sort Journal Data	3.10.5	47

1
2 Not required if DBK2 is a VSAM file
3 Not required unless DBK2 is a VSAM file

REORG	RSTU	JOBNAME	STEPNAME	Description		Page
			DBEZCUP2	Generate DML	3.10.6	48
			DBEZSDML	Sort DML	3.10.7	48
			DBEZSDBK	Sort DBKeys	3.10.8	48
			DBEZSXT3	Add new DBKey data	3.10.9	48
			DBEZCUP3	Apply DML	3.10.10	49
			DBEZSXT4[1]	Sort Dbkeys	3.10.11	49
			DBEZCOPY	Merge DBKeys	3.10.12	49
				Backup DBZCNTL and “carried-forward” files	3.10.13	50
✓	✓	DBEZRFWD	All	Pre-Implementation - Final Journals	3.11	51
✓	✓			Database Implementation	3.12	52

1 Not executed if using VSAM DBEZDBK2 file

3.1 INSTDBEZ – Allocate Work Files

This base install step is run to allocate all of the runtime work files used by the subsequent steps/processes.

Note, you must not change the DCB details from those specified in INSTDBEZ.

The following datasets are pre-allocated (If you intend to use a VSAM file for DBZDBK2 then use the DD DUMMY version of its DD statement):

DATASET	DESCRIPTION
prefix.DBEZCNTL	The Syncsort IDMS Reorg control file.
prefix.RECPGXRF	The recid/page group cross reference file to be initialised.
prefix.RECPGXRF.SORT	The sorted recid/page group cross reference file.
prefix.UCFBATCH	The UCF Batch file containing generated DCMT VARY AREA commands.
prefix.IDMSLDEL	The logically deleted records file.
prefix.DBEZSR23	The SR2/SR3 file.
prefix.DBEZSR23.SORT	The sorted SR2/SR3 file.
prefix.DBEZDBK2	The master “carried-forward” DBKey file – dummy this file if you intend to use a VSAM DBK2 file.
prefix.DBEZDBK3	Intermediate DBKey file.
prefix.DBEZDBK4	Intermediate DBKey file.
prefix.DBEZDBU2	Sorted RSTU keys file.
prefix.DBEZRUPD	Run-unit ID file.
prefix.DBEZRUPD.SORT	Sorted run-unit ID file.
prefix.DBEZCFRU	The “carried-forward” run-unit file.
prefix.DBEZRTSV	RTSV file.
prefix.DBEZDML	Catch-up DML to apply file.
prefix.DBEZDML.SORT	Sorted catch-up DML to apply file.
prefix.DBEZJNLU	Journal updates file.
prefix.DBEZJNLU.SORT	Sorted journal updates file.
prefix.DBEZCFJN	The “carried-forward” journal file.

3.2 DBEZJOB0 - Initialise Work Files

This preliminary step is run to initialise DBZSR23, RECPGXRF and DBZDBK2 and the “carried-forward” files (DBZCFRU, DBZCFJN) which must be empty for the first invocation of DBEZRFWD.

The JCL includes the following datasets :-

FILENAME	DATASET	DESCRIPTION
RECPGXRF	prefix.RECPGXRF	The recid/page group cross reference file to be initialised.
DBZSR23	prefix.DBEZSR23	The SR2/SR3 file to be initialised.
DBZCFRU	prefix.DBEZCFRU	The “carried-forward” run-unit file to be initialised.
DBZCFJN	prefix.DBEZCFJN	The “carried-forward” journal file to be initialised.
DBZDBK2	prefix.DBEZDBK2	The “carried-forward” DBKey file to be initialised – dummy this file if you intend to use a VSAM DBK2 file.

Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg/RSTU run-time files. Refer to the INSTDBEZ job for further information.

Note: The job contains an optional step to allocate a VSAM file for the DBZDBK2 file if you choose to use a VSAM dataset for this file. See also the DBZDBK2=VSAM parameter in the next section.

Sample JCL for this job can be found in the Appendix.

3.3 DBEZJOB1 - Create Syncsort IDMS Reorg/RSTU Control File

This preliminary step is run to allocate and initialise the Syncsort IDMS Reorg/RSTU control file (DBZCNTL). Syncsort IDMS Reorg/RSTU will use this file to record control information required for RESTARTS and journal sequence numbers for sequence checking.

The input parameters are stored in the DBZCNTL file. In addition, the program will produce a report to the SYSLST output listing the DMCL and subschema name, the segments/areas being processed, and the dependant areas with a flag for the presence of records stored DIRECT.

The Record ID/Page Group cross reference file is initially populated with details from the 'old' subschema/dmcl. This file is then sorted and processed again to add the new page group details, using the new subschema/dmcl. This cross reference file is used find the new page group details when applying DML updates to the reorg'd database. This allows Syncsort IDMS Reorg to process mixed page groups in a single run and support mixed page group sets that were introduced in CA IDMS R18.5.

After old and new details have been added to the RecID Cross Reference file and report lists the record ID's from the old subschema, with the old and new segement/page group details.

If you do have records stored DIRECT, then you may need to make special provision for these through a user supplied exit. Details of how to do this are provided in the Technical Notes section *4.1**.*

Sample JCL for this job can be found in the Appendix.

SYSIPT Parameters

Input parameters to this job are supplied through the SYSIPT file and also through a SYSIDMS file. The SYSIPT parameters are described below. Note that each parameter must start on a new line, though not necessarily in column one. Comments may be added to each line if preceded by an asterisk.

The following parameters can be specified in SYSIPT:

```
PROCESS=REORG/INDEX/RSTU
SUBSCHEMA=old-subschema-name
NEWSUBSCHEMA=new-subschema-name
DBNAME=dbname
COMMIT=YES/NO
DBEZDBK2=VSAM
SEGMENT=segment-name
NEWSEGMENT=new-segment-name
DMCL=dmcl-name
RSTTMOD=rstt-modname
DBEZEXT1=exitname-1
DBEZEXT2=exitname-2
AREA=area-name-1
AREA=area-name-2 etc.
NEWAREA=area-name-1
NEWAREA=area-name-2 etc.
```

All of the parameters can be input in any order, except for the AREA parameter statements, which must be the last statements specified. For example, the following is an example of a valid SYSIPT;

```
PROCESS=REORG
SUBSCHEMA=SSDB001A
SEGMENT=SEGMAIN
COMMIT=NO
AREA=DB001A-AR-DF001
AREA=DB001A-AR-DF002
```

The parameter syntax is now described in detail.

PROCESS=REORG/INDEX/RSTU

PROCESS=REORG is for database reorganisation (Unload/Reload or DB-Reorg equivalent).

PROCESS=INDEX is for index re-builds.

PROCESS=RSTU is for database restructure.

To process a reorg and restructure in the same run, specify two parameters, namely PROCESS=REORG and PROCESS=RSTU.

The following combinations of PROCESS are supported in the same run:-

PROCESS=REORG+RSTU	✔
PROCESS=INDEX +RSTU	✔
PROCESS=INDEX +REORG	✘

Refer to the Tech Note in section 4.9 for further information about combining multiple PROCESS statements in a single run.

SUBSCHEMA=subschema-name

Subschema-name specifies the subschema which describes the database to be unloaded/restructured.

By default, if a subschema-name is not specified, it indicates that an SQL-type segment is to be processed. In this case, Syncsort IDMS Reorg/RSTU will build the required subschema.

NEWSUBSCHEMA=new-subschema-name	New-subschema-name specifies the subschema that describes the segment in which the data will be reloaded (or the restructured data). By default, if new-subschema-name is not specified, the subschema that was used to unload the data will be used by DBEZRFWD. If subschema-name was not specified (ie., an SQL-type segment is to be reloaded) Syncsort IDMS Reorg will build the required subschema used by DBEZRFWD. *There is an additional consideration with regard to the OLD and NEW Subschema names. That is, <u>if any database procedures have changed</u>, the DBEZRFWD job will need access to both OLD and NEW subschemas via STEPLIB. This means that the OLD and NEW subschema names must be different.*
COMMIT=YES/NO	COMMIT=YES is the default value and means that DBEZCUP3 will issue COMMITs every 10,000 DML requests, and local mode journaling is therefore required to enable recovery from a failure during DBEZCUP3 without going back to the start. COMMIT=NO means that DBEZCUP3 will not issue COMMITs and therefore local mode journals are not required. In this case, you are advised to backup the database files (affected areas) at the end of each iteration of DBEZRFWD. Refer to section 4.3 for further details about DBEZRFWD recovery considerations.
AREA=area-name	area-name specifies the name of an area to be unloaded or restructured. You must code one such statement for each area input to IDMSUNLD, or to RESTRUCTURE, or the area owning each set for an INDEX rebuild.
NEWAREA=area-name	If a new area is being created, e.g., to hold an index set, and an UNLOAD/RELOAD being performed to populate it, then you must specify NEWAREA=area-name where area-name specifies the name of the area which exists in the “new” SUBSCHEMA that does not exist in the “old” SUBSCHEMA.
RSTTMOD=rstt-modname	rstt-modname specifies the name of the base restructuring table. This is the same table as that being used by the RESTRUCTURE process. Refer to the CA IDMS utilities manual for further details.
DBEZEXT1=exitname-1	The name of the user supplied exit which may be required for STORE DIRECT. Refer to Technical Notes section 4.1 for further details.
DBEZEXT2=exitname-2	The name of the user supplied exit to be called by DBEZRFWD process prior to STORE or MODIFY but after call to IDMSRSTU. Refer to Technical Notes section 4.2 for further details.

SEGMENT=segment-name

segment-name is the name of the database segment to be unloaded (or restructured).

NEWSEGMENT=new-segment-name

new-segment-name specifies the segment in which the data will be reloaded. This parameter will default to the old segment-name if not specified.

DMCL=dmcl-name

dmcl-name specifies the DMCL that defines the segment to be unloaded (or restructured).

By default, if you do not specify dmcl-name, the DMCL to be used is taken from the SYSIDMS parameter file. If you did not specify a DMCL in the SYSIDMS parameter file, the IDMSDMCL name is used.

The DMCL that defines the segment the data will be reloaded into, is defined later in SYSIDMS for the DBEZRFWD step.

In particular circumstances, related to the use of database procedures, it is required that both the old and new segments are available in the DMCL specified in the SYSIDMS for DBEZRFWD. To simplify matters, the recommendation now is to use the same DMCL containing the unload and reload segments (where different) both here and in the DBEZRFWD step.

Refer to the technical notes section for further information.

DBEZDBK2=VSAM

Specify this optional parameter if you are using a VSAM file for the DBEZDBK2 file.

The advantage of a VSAM file is that DBEZSXT3 and DBEZSXT4 will run much quicker and DBEZCOPY is no longer executed. However, the disadvantage is that DBEZDBK2 becomes bigger because it is VSAM and requires free space.

Note: If using a VSAM dataset for this file, it is allocated by the optional job-step of DBEZJOB0.

Specifying SYSIDMS Parameters.

You should use SYSIDMS to provide the details of the DMCL to use and if applicable DBNAME (Release 18.5+ Reorg by DBName).

Step	DMCL	DBNAME
DBEZCNTL	Specify the DMCL used for the UNLOAD/REORG of the 'old' database/segment.	If running DB-REORG by DBNAME (CA IDMS R18.5+), then you must specify the **'old' DBNAME** that specifies the original database.
DBEZXREF	Specify the DMCL used for the reload of the 'new' database/segment.	If running DB-Reorg by DBNAME (CA IDMS R18.5+), then you must specify the **'new' DBNAME** that specifies the reloaded database.

SQL Only; In addition, for SQL ONLY, DBEZJOB1 requires access to the catalog containing the schema definition for the applicable SQL database areas. The following SYSIDMS parameter may therefore be required:-

DBNAME=dbname.

where dbname identifies the database containing the applicable catalog area. This parameter is not required if the applicable catalog is contained within the system default database.

You will also need to include the file assignments for the applicable catalog areas in the JCL unless dataset name information is specified in the (DMCL) file definitions.

The JCL includes the following datasets:-

FILENAME	DATASET	DESCRIPTION
STEPLIB	EZ-DB.loadlib caidms.old.dbaload	The dataset name of the load-library containing the Syncsort IDMS Reorg/RSTU run-time modules.
DBZCNTL	prefix.DBEZCNTL	The dataset name of the DBZCNTL file. The file will be initialized as required. This file is used in subsequent job steps.
RECPGXRF	prefix.RECPGXRF	The dataset name of the RecID/Page Group cross reference file. This file is used in subsequent job steps.
UCFBTCH	prefix.UCFBATCH	The dataset name of the UCFBATCH file. DBZCNTL will output a DCMT VARY AREA statement to this file for each area affected by the Unload/Reload process. That is, the areas being re-sized, and areas with dependencies created by cross-area pointers. This file can be used at the start of the “Inflight Backup” step described later.

Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg/RSTU run-time files. Refer to the DBEZALOC job in the appendix for further information.

DBEZCNTL outputs an end of step report to SYSLST as follows:-

```
********************************* TOP OF DATA **********************************
16 JUL 14 16:01        SYNCSORT IDMS REORG AREA REPORT                     PAGE   1
           DMCL: DMGLBD2M  SUBSCHEMA: MPGIXSUB   DBNAME: MPGDB003
AREA NAME                     PGGP  DPNDNT  DIR DBKY
MPGSEG1 .AREA0001             0500
MPGSEG2 .AREA0002             0600
MPGSEG2 .AREA0003             0600    Y
MPGSEG4 .AREA0004             0450
MPGSEG5 .AREA0005             0550
MPGSEG6 .AREA0006             0650             Y
******************************** BOTTOM OF DATA ********************************
```

Figure 2 DBEZCNTL SYSLST Report

Use the report to identify dependent areas and areas containing Direct DBKeys.

All of the areas included in the RESTRUCTURE and/or UNLOAD/RELOAD process AND any additional areas designated as “dependent” must be available to the DBEZRFWD process.

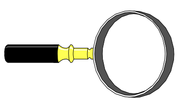
DBEZXREF outputs an end of step report to SYSLST as follows:-

```
********************************* TOP OF DATA **********************************
16 SEP 14 16:01       SYNCSORT IDMS REORG RECORD XREF REPORT                PAGE   1
           DMCL: DMGLBD2M  OLD SS: MPGIXSUB  OLD DBN: MPGDB003
                           NEW SS: MPGIXSUB  NEW DBN: MPGDBX03
RECID  ORIG SEG ORIG PG RADIX  NEW SEG   NEW PG RADIX
 9100  MPGSEG1      500  FF08   MPGSEG1X    500  7F07
 9200  MPGSEG1      500  FF08   MPGSEG1X    500  7F07
 9300  MPGSEG2      600  FF08   MPGSEG2X    600  7F07
 9400  MPGSEG4      450  FF08   MPGSEG4X    460  FF08
 9500  MPGSEG5      550  FF08   MPGSEG5     550  FF08
 9600  MPGSEG4      450  FF08   MPGSEG4X    460  FF08
 9700  MPGSEG6      650  FF08   MPGSEG6X    660  FF08
******************************** BOTTOM OF DATA ********************************
```

Figure 3 DBEZXREF SYSLST Report

3.4 Create database for Restructure/UNLOAD

The operation of Syncsort IDMS Reorg/RSTU relies on the fact that you will Restructure or Reorg/UNLOAD/RELOAD a copy of the database, while the original database is still available to the production system.



The copy should include the areas being restructured and/or reorganised, that is, the areas included in the RESTRUCTURE and/or UNLOAD procedure(s) and DBEZJOB1 input stream, as well as any areas designated by DBEZJOB1 as being dependent. You do not need to copy the entire database.

This section describes how to create a such a copy.

A quiesce point must normally be established at the start point of the re-size process. It is possible to minimise the loss of service to the end-user during this step through the use of Inflight backup techniques.

There are a variety of Inflight Backup procedures in common use. The following steps outline one possible sequence designed to create a copy of the areas affected by the re-sizing operation, but minimising the down-time at this stage of the procedure.

Note: It is not mandatory that you create the copy using an Inflight Backup technique. You may prefer to create the back-up with the affected areas offline, if this is the standard procedure at your site. However, Inflight Backup would be used in situations where you are aiming to minimize the overall down-time.

3.4.1 Inflight Backup Procedure with Quiesce point

1. Vary all areas affected by the Restructure and/or unload/reload process to retrieval. This includes the areas being re-sized and all the areas with dependencies created by cross-area pointers.

   For this purpose you can use the UCFBTCH file created by the DBEZCNTL job

2. Vary Journal. The **next** offloaded journal (after the one off-loaded as the result of this Vary Journal) will be the first journal fed into the Syncsort IDMS Reorg/RSTU "Catch-Up" phase
3. Issue **DBEZSTRT** command. Refer to section for details.
4. Vary affected areas back to update.
5. Take a secure copy of the affected areas.
6. Vary journal again.
7. Concatenate all Journals produced from (4) through (6) to a single file. Do not include the journal archive produced at (2). For sites using data sharing, where journals are merged from multiple CV's, refer to 5.4 – Merged Journal Option.
8. Create a copy on DASD of the affected areas using the backup taken at (4).
9. Using the concatenated journal, run Roll-back against the restored copy of the secured areas.
10. Optionally, take a secure copy of the roll-backed areas. This copy may be used in the event of problems in the Unload/Reload process requiring a restart at the Unload step. This would save having to rerun the Roll-back process to recreate a clean set of areas.

3.4.2 Writing the DBEZSTRT quiesce-point record to the journal

The in-flight backup technique described above identifies the first archived journal to be processed by the Syncsort IDMS Reorg/RSTU “catch-up” process. It is obviously critical that the subsequent “catch-up” processing should begin at the correct point. Therefore, to avoid the possibility of a user inadvertently commencing the “catch-up” with the wrong journal file, a program is provided that will write a USER record to the journal file for subsequent validation.

After Varying the affected areas to retrieval and Varying the journal file (to create the required quiesce point) the user must issue the **DBEZSTRT** command from a DC/UCF enter next task code prompt (or UCFBATCH job) to indicate that a Syncsort IDMS Reorg/RSTU process is commencing.

DBEZSTRT will write a USER record to the journal file, and output the following message to the operator and MVS log:

DB-EZR STAMP WRITTEN TO JOURNAL FILE

When commencing the “catch-up” process, DBEZRFWD should detect this USER record prior to any journal records for any affected area(s). Refer to section 3.10.1 for a description of how DBEZRFWD uses this record to ensure that the “catch-up” has commenced from the correct place.

3.5 DBEZJOB2 Prepare for “Catch Up”

There are various “house-keeping” functions which must be completed before the Syncsort IDMS Reorg “catch-up” phase is commenced.

3.5.1 DBEZSR23 - Identify SR2/SR3 DBKeys

The purpose of this step is to identify any SR2/SR3 pairs for records or record fragments relocated as a result of previous restructures. This step should be run before the database unload and/or restructure.

This process runs program DBEZSR23 to read all of the pages from the copy of the areas affected by the Unload/Reload. It will scan the line-index of every page searching for SR2 records. If an SR2 is found, then its DBKey and the records content are written out to the output file - DBZSR23. (The record content is the DBKey of SR3 and record Id of the original record).

In addition, DBEZSR23 will search for Logically Deleted records and, if present, will write the applicable DBKeys of the Logically Deleted records to the output file IDMSLDL.

Before you can UNLOAD/RELOAD a database, you must first physically erase logically deleted records using BCF CLEANUP.

FILENAME	DATASET	DESCRIPTION
STEPLIB	EZ-DB.loadlib	The dataset name of the load-library containing the Syncsort IDMS Reorg run-time modules.
	caidms.old.dbaload	The dataset name of the load-library containing the applicable CA-IDMS Subschema and DMCL Load-modules currently in use in the Production system. ie. the old subschema and DMCL.
DBZCNTL	prefix.DBEZCNTL	The dataset name of the DBZCNTL file.
DBZSR23	prefix.DBEZSR23	The dataset name to which DBEZSR23 will write the SR2/SR3 data.
IDMSLDL	prefix.IDMSLDEL	The dataset name to which DBEZSR23 will write the Logically Deleted Record data. Equivalent to SYSPCH from IDMSDUMP utility.
userdb	user.userdb.copy	**userdb** is the DD name of the applicable CA-IDMS file. Include one such statement for each and every file included in the affected area(s). Note that this job-step must be run against the COPY of the affected area(s) against which you will run the reorg. **user.userdb.copy** is the dataset name corresponding to this CA-IDMS file. Note previous comment that this is the COPY of the affected area against which you will run the reorg.

Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg run-time files. Refer to the INSTDBEZ job in the appendix for further information

DBEZSR23 writes an end of job report to SYSLST as follows:-

```
********************************* TOP OF DATA **********************************
16 AUG 14 15:59       SYNCSORT IDMS REORG SR2/SR3 AREA REPORT               PAGE   1
           DMCL: DMGLBD2   SUBSCHEMA: SSEZRE10  SEGMENT: EZREORG
AREA NAME                    SR2 COUNT  LDR COUNT
EZREORG .EZREORG-AREA1               1          0
EZREORG .EZREORG-AREA2               0      1,056
EZREORG .EZREORG-AREA3              12          0
EZREORG .EZREORG-AREA4             134        145
******************************** BOTTOM OF DATA ********************************
```

Figure 4 DBEZSR23 SYSLST Report

3.5.2 SORTSR23 - Sort DBZSR23

The output file from the previous step is sorted in ascending sequence of DBKey.

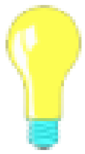
Sample JCL for this job can be found in the Appendix. The JCL includes the following datasets:-

FILENAME	DATASET	DESCRIPTION
SORTIN	prefix.DBEZSR23	The dataset name containing the SR2/SR3 data.
SORTOUT	prefix.DBEZSR23.SORT	The dataset name to which the SORTed SR2/SR3 data is written.

Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg run-time files. Refer to the INSTDBEZ job in the appendix for further information

3.6 Perform Database Restructure

Perform the database restructure against the copy areas using the same procedure as you would in the absence of Syncsort IDMS RSTU . For example, using the BCF RESTRUCTURE SEGMENT and BCF RESTRUCTURE CONNECT.

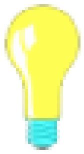


You cannot use the RESTRUCTURE SEGMENT utility statement to make changes that will modify the prefix portion of a record if there are logically deleted records in the database area to be modified. To ensure there are no logically deleted records, use the CLEANUP utility (against the copy) to erase them before performing a restructure operation.

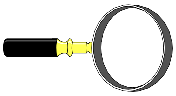
Refer to the CA IDMS Utilities manual for full details.

3.7 Perform Unload/Reload

Run the standard procedures to Unload/Reload the areas being re-sized. This should, as usual, be preceded by a BCF CLEANUP, and when complete, a secure copy of the re-sized area(s) taken.



The UNLOAD may be commenced in parallel with the previous step, that is SORTSR23. It can start as soon as the DBEZSR23 step has completed.



*DBEZJOB3 sorts the file of DBKeys produced by the IDMSDBL2 step of the reload process in SYS005. To preserve this file of DBKeys for DBEZJOB3 you must **not** specify AS SORTEXIT or REUSE WORKFILES. If either option is specified, then SYS005 will not contain the DBKeys. You may however step through the Unload/Reload until the IDMSDBL2 step is complete, take a copy of SYS005 and then run the remaining steps using these options.*

3.8 DBEZJOB3 Prepare for “Catch UP” (non-VSAM DBEZDBK2)

If using a VSAM DBZDBK2 file, refer to DBEZJOB4 instead of DBEZJOB3.

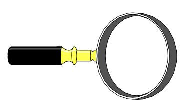
There are **alternative** versions of this job which must be run dependent on whether you are using UNLOAD/RELOAD or DB-Reorg for the database reorganisation process.

- For **UNLOAD/RELOAD** use job DBEZJB3U
- For **DB-REORG** use job DBEZJB3R

3.8.1 DBEZJB3U - Generate file of old/new DBKeys from IDMSDBL2

This job sorts the output file produced by IDMSDBL2 and may be run at any time after IDMSDBL2 has completed. The file is sorted into OLD DBKey sequence with sort exit DBEZSXT8 stripping out all data other than the OLD/NEW DBKey pairs, where the OLD DBKey is from one of the affected areas. It also adds the New Page Group details.

Sample JCL for this job can be found in the Appendix. The JCL includes the following datasets:-

FILENAME	DATASET	DESCRIPTION
SORTIN	EZ-DB.dbl2out	The DBKey file output by IDMSDBL2.  *This file is the SYS005 output file produced by IDMSDBL2.* *You must ensure that your Unload/Reload JCL is set up to retain this output file after completion. Some sites include a job-step to automatically delete the work files. Refer to the CA-IDMS Utilities manual for further information.*
DBZDBK2	prefix.DBEZDBK2	The Sort Extract File of Old/New DBKeys.
RECPGXRF	prefix.RECPGXRF.SORT	The Sorted RecID/Page Group cross reference file.
DBEZLIB	EZ-DB.loadlib	The dataset name of the load library containing the Syncsort IDMS Reorg run-time load-modules.

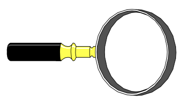
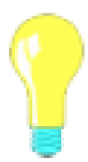
Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg run-time files. Refer to the INSTDBEZ job in the appendix for further information

3.8.2 DBEZJB3R - Generate file of old/new DBKeys using DB-Reorg

This step sorts the output file produced by DB-Reorg and may be run at any time after DB-Reorg has completed. The file is sorted into OLD DBKey sequence with sort exit DBEZSXT6 stripping out all data other than the OLD/NEW DBKey pairs, where the OLD DBKey is from one of the affected areas. It also adds the New Page Group details.

The DBKey file is then further sorted by Page Group and DBKey to cater for DB-Reorg by Database.

Sample JCL for this job can be found in the Appendix. The JCL includes the following datasets:-

FILENAME	DATASET	DESCRIPTION
SORTIN	EZ-DB.dbreorg.dbkeys	The DBKey files output by DB-Reorg.  *These files are the DBKEYS output files produced by DB-Reorg. These files (there may be more than one) are only produced if you have specified 'GENERATE DBKEYS' on the RELOAD step of DB-REORG.* *If there is more than one file generated then concatenate them together for SORTIN.*
DBZDBK3	prefix.DBEZDBK3	The Sort Extract File of Old/New DBKeys.
DBZDBK2	prefix.DBEZDBK2	The Old/New DBKeys file, sorted on Page Group & DBKey.
RECPGXRF	prefix.RECPGXRF.SORT	The Sorted RecID/Page Group cross reference file.
DBZSR23	prefix.DBEZSR23	The dataset name to which DBEZSXT6 will write the SR2/SR3 data.  *If you have performed a restructure prior to the DB-Reorg, you should use the DBZSR23 file produced by DBEZJOB2 in the catch-up phase. Therefore, this file should be dummied out of the DBEZSXT6 job-step.*
DBEZLIB	EZ-DB.loadlib	The dataset name of the load library containing the Syncsort IDMS Reorg run-time load-modules.

Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg run-time files. Refer to the INSTDBEZ job in the appendix for further information.

3.9 DBEZJOB4 Prepare for “Catch UP” (VSAM DBEZDBK2)

If using a non-VSAM DBZDBK2 file, refer to DBEZJOB3 instead of DBEZJOB4.

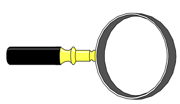
There are **alternative** versions of this job which must be run dependent on whether you are using UNLOAD/RELOAD or DB-Reorg for the database reorganisation process.

- For **UNLOAD/RELOAD** use job DBEZJB4U
- For **DB-REORG** use job DBEZJB4R

3.9.1 DBEZJB4U - Generate file of old/new DBKeys from IDMSDBL2

This job sorts the output file produced by IDMSDBL2 and may be run at any time after IDMSDBL2 has completed. The file is sorted into OLD DBKey sequence with sort exit DBEZVXT8 stripping out all data other than the OLD/NEW DBKey pairs, where the OLD DBKey is from one of the affected areas. It also adds the New Page Group details.

Sample JCL for this job can be found in the Appendix. The JCL includes the following datasets:-

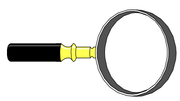
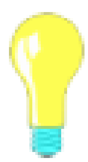
FILENAME	**DATASET**	**DESCRIPTION**
SORTIN	EZ-DB.dbl2out	The DBKey file output by IDMSDBL2.  *This file is the SYS005 output file produced by IDMSDBL2.* *You must ensure that your Unload/Reload JCL is set up to retain this output file after completion. Some sites include a job-step to automatically delete the work files. Refer to the CA-IDMS Utilities manual for further information.*
DBZDBK2	prefix.DBEZDBK2	The Sort Extract File of Old/New DBKeys.
RECPGXRF	prefix.RECPGXRF.SORT	The Sorted RecID/Page Group cross reference file.
DBEZLIB	EZ-DB.loadlib	The dataset name of the load library containing the Syncsort IDMS Reorg run-time load-modules.

Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg run-time files. Refer to the INSTDBEZ job in the appendix for further information

3.9.2 DBEZJB4R - Generate file of old/new DBKeys using DB-Reorg

This step sorts the output file produced by DB-Reorg and may be run at any time after DB-Reorg has completed. The file is sorted into OLD DBKey sequence with sort exit DBEZVXT6 stripping out all data other than the OLD/NEW DBKey pairs, where the OLD DBKey is from one of the affected areas. It also adds the New Page Group details.

Sample JCL for this job can be found in the Appendix. The JCL includes the following datasets:-

FILENAME	DATASET	DESCRIPTION
SORTIN	EZ-DB.dbreorg.dbkeys	The DBKey files output by DB-Reorg.  *These files are the DBKEYS output files produced by DB-Reorg. These files (there may be more than one) are only produced if you have specified 'GENERATE DBKEYS' on the RELOAD step of DB-REORG.* *If there is more than one file generated then concatenate them together for SORTIN.*
DBZDBK2	prefix.DBEZDBK2	The Old/New DBKeys file.
RECPGXRF	prefix.RECPGXRF.SORT	The Sorted RecID/Page Group cross reference file.
DBZSR23	prefix.DBEZSR23	The dataset name to which DBEZSXT6 will write the SR2/SR3 data.  *If you have performed a restructure prior to the DB-Reorg, you should use the DBZSR23 file produced by DBEZJOB2 in the catch-up phase. Therefore, this file should be dummied out of the DBEZSXT6 job-step.*
DBEZLIB	EZ-DB.loadlib	The dataset name of the load library containing the Syncsort IDMS Reorg run-time load-modules.

Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg run-time files. Refer to the INSTDBEZ job in the appendix for further information.

3.10 DBEZRFWD - “Catch-up” Phase

The Syncsort IDMS Reorg/RSTU “catch-up” phase is performed by the Job DBEZRFWD. This consists internally of multiple steps which are described later in this section. It is necessary to be aware of the function of these job steps, as it is may be necessary to RESTART this job in certain circumstances.

The “Catch up” processing performed by DBEZRFWD is, in essence, the duplexing of updates performed against the PRODUCTION database against the COPY database. This is performed by processing every journal tape off-loaded from the production system **since the start of the Syncsort IDMS Reorg/RSTU process**, that is, after the quiesce point established prior to creating the copy database. Refer to section 3.4 for further details.

The journal tapes must be processed in chronological sequence. Batches of Journal tapes may be concatenated together in the JCL so long as they are concatenated in the correct sequence. When the term “concatenated “ is used in the context of DBEZRFWD, we refer to JCL concatenation only. It is NOT necessary for the files to be physically concatenated.

For sites using Data Sharing with journals from multiple CV’s, you need to use the Syncsort IDMS Reorg Merged Journal Option. Refer to 5.4 - Merged Journal Option for further details.

In practice, it is likely that execution of this job will be an iterative procedure. The first partial execution may be performed in parallel with the later stages of the database reload. Subsequent iterations will be submitted until all off-loaded journals have been processed. Refer to 3.10 later for the procedure for processing the final “active” journal.

Any number of Journal files may be submitted in a single run. However, it is recommended that the number of Journal files to be processed by a single iteration of DBEZRFWD should be limited, as this will allow SORT work-files etc. to be on DASD, thus greatly speeding up the process. Initially, we suggest that you limit the number of journals for each run to, say 3-5, depending on the size and expected number of updates to the affected areas.

3.10.1 Validating the DBEZSTRT quiesce-point record

You must use the DBEZSTRT function (as described in section 3.4.2) to write a user record to the journal file in order that the DBEZRFWD job can validate that you have begun the “catch-up” from the correct archived journal. Only the first DBEZSTRT User record is used to check the start point. If you have subsequent DBEZSTRT User records (for example if you run DBEZSTRT as a startup autotask and have cycled CV multiple times during the catch up phase) then they are not used. The user journal record is used as follows:

When commencing the “catch-up” process, DBEZRFWD should detect this USER record prior to any journal records for any affected area(s). DBEZCUP1 (the first step in the DBEZRFWD job) will therefore check for the USER record and if found, will output the following message:

DB905526 JOURNAL STAMP JSEQ nnnnnnnnnn

where nnnnnnnnnn is the journal sequence number of the USER record. If it does not find the USER record, and then detects a journal record which updates one of the affected areas, it will output the following messages:

DB905527 DBEZ UPDATES BEFORE JOURNAL STAMP
DB905528 DBEZ WILL TERMINATE AFTER DBEZCUP1

DBEZRFWD will then terminate after completion of DBEZCUP1.

If you have inadvertently submitted the wrong journal file into DBEZRFWD, you should re-start DBEZRFWD with the correct archived journal after first re-running the DBEZCNTL job. This is required in order to re-initialize the carried-forward files. Do not in this case specify RESTART=DBEZSUPD.

3.10.2 DBEZRFWD input parameters

Input parameters to the DBEZRFWD step may be specified through an EXEC PARM statement and also through a SYSIDMS file.

The EXEC parameters which may be specified are as follows:-

PARM='DBNAME=dbname,RESTART=**stepname-1**,STOPAFTER=**stepname-2'**

All of the parameters are optional.

The DBNAME parameter allows the user to overide the DBNAME parameter specified in DBEZJOB1.

dbname Identifies the name of the database to BIND to at run-time in DBEZCUP3 and overrides the dbname parameter specified in DBEZJOB1. Refer to the DBEZJOB1 DBNAME parameter for more information. If specified here, this parameter must be an 8-character alpha-numeric string. That is, if the name is less than 8-characters, you must pad with spaces. eg.,

PARM='DBNAME=USERDB '

In order to permit the various internal phases (steps) to be selected and executed individually, you may specify the optional RESTART and/or STOPAFTER parameters. Where,

stepname-1 is the first step to be executed

stepname-2 is the last step to be executed.

Valid step names which may be specified for stepname-1 and stepname-2 are as follows:-

DBEZCUP1
DBEZSUPD
DBEZSJNL
DBEZCUP2
DBEZSDML
DBEZSDBK
DBEZSXT3
DBEZCUP3
DBEZSXT4
DBEZCOPY

If no EXEC PARM is specified, DBEZRFWD will commence at DBEZCUP1 and continue with all steps through DBEZCOPY, except in the case of a restart after a failure, in which case DBEZRFWD will work out itself (using the DBZCNTL file) where to continue from. Refer to section 4.3.2 for further details about restarting DBEZRFWD in the event of an error.

3.10.2.1 Specifying SYSIDMS parameters

SYSIDMS may contain optional input parameters for DMCL and DBNAME name, as follows:-

DMCL=dmcl-name
DBNAME=new-dbname

Where dmcl-name is the name of the DMCL load module to be used. If you do not specify this parameter, then the system default DMCL name will be used.

New-dbname must be specified if you are processing a DB-Reorg run be DBName and must specify the new DBName for DBEZRFWD to bind against.

SQL Only; In addition, for SQL ONLY, DBEZRFWD requires access to the catalog containing the schema definition for the applicable SQL database areas. The following SYSIDMS parameter may therefore be required:-

DBNAME=dbname.

where dbname identifies the database containing the applicable catalog area. This parameter is not required if the applicable catalog is contained within the system default database.

You will also need to include the file assignments for the applicable catalog areas in the JCL unless dataset name information is specified in the (DMCL) file definitions.

Sample JCL for this job can be found in the Appendix.

The JCL includes the following datasets:-

FILENAME	**DATASET**	**DESCRIPTION**
STEPLIB	EZ-DB.loadlib	The dataset name of the load-library containing the Syncsort IDMS Reorg/RSTU run-time modules.
	caidms.new.dbaload	The dataset name of the load-library containing the applicable CA-IDMS Subschema and DMCL Load-modules Note: This must be the library containing the NEW Subschema and DMCL modules, that is, for the re-sized area(s). *DBEZCUP2 requires the OLD subschema load-module in order to access information relating to database procedures.* *If OLD and NEW subschema names are the same, and there are no changes to database procedures, then the NEW subschema may be used by DBEZCUP2. This will usually be the case.* *If there are changes to database procedures, then OLD and NEW subschema names must be different, and the OLD subschema must also be accessable via this library.*
	caidms.loadlib	The dataset name of the library containing the standard CA-IDMS executable load modules.

FILENAME	DATASET	DESCRIPTION
DBZCNTL	prefix.DBEZCNTL	The dataset name of the DBZCNTL file.
DBZCFJN	prefix.DBEZCFJN	Journal records for incomplete success units to be fed into the next run of this process. ie., the “Carried-Forward” file. This will be empty for the first run.
DBZCFRU	prefix.DBEZCFRU	Open Run-Units to “Carry Forward” to next run. This will be empty for the first run.
DBZRTSV	prefix.DBEZRTSV	Required by SQL.
DBZRUPD	prefix.DBEZRUPD	Successful Run-Unit Ids.
DBZRUPS	prefix.DBEZRUPD.SORT	DBZRUPD sorted by ascending run-unit id.
DBZDML	prefix.DBEZDML	A DML File. This contains all of the required DML commands and data to replicate all of the updates to the affected areas and a sequence number to facilitate restarts.
DBZDMLS	prefix.DBEZDML.SORT	DBZDML is sorted by the DML sequence number within the last journal sequence number for the DML to ensure the DMLs are applied in the order in which they completed.
DBZSR23	prefix.DBEZSR23.SORT	File containing details of SR2/SR3 records.
DBZDBK2	prefix.DBEZDBK2	The Sort Extract File of Old/New DBKeys.
DBZDBK3	prefix.DBEZDBK3	A file of DBKeys required to perform the DML.
DBZDBK4	prefix.DBEZDBK4	A file of DBKeys required to perform the DML.
RECPGXRF	prefix.RECPGXRF.SORT	The Sorted RecID/Page Group cross reference file.
DBZJNLU	prefix.DBEZJNLU	Journal records for the affected areas to pass forward to the next step.
DBZJNLS	prefix.DBEZJNLU.SORT	DBZJNLU sorted by ascending journal sequence number within run unit id.

FILENAME	DATASET	DESCRIPTION
IDMSJNL	idms.journal.file-1,... idms.journal.file-n	The archived journal files from the production system. You may include several such files in each iteration of DBEZRFWD provided they are “concatenated” in chronological sequence. *Concatenated” in this context refers to JCL concatenation. The files do not have to be physically concatenated.* Do not included too many journal files in a single iteration as this will affect the size of intermediate sort work files etc. For Data Sharing see 5.4 - Merged Journal Option.
SYSJRNL	prefix.SYSJRNL	Local Mode Journal Files. If you have specified COMMIT=YES (which is the default) you must include local mode journals through this DD statement. *It is recommended that you allocate journal files as per your central version allocations, and assign them to a GDG in order to simplify the process of restarts etc.*
RBCKJNL		If you are restarting DBEZCUP3 after a manual recovery, and you are using local mode journals (and COMMIT=YES) you must include the local journals used for the roll-back in the next iteration of DBEZRFWD as DBEZRFWD will use these journals to locate the previous Commit point. Refer to section 4.3.2 for further information about DBEZRFWD recovery.
Userdb	user.userdb.copy	userdb - The file name (DDname) of a file included in an area affected by the RESTRUCTURE and/or UNLOAD/RELOAD process. For UNLOAD/RELOAD, this file is the RELOADED file to which this process is now going to “catch-up” the updates issued against the production database (the old file). You must include one such DD statement for each applicable file.

FILENAME	DATASET	DESCRIPTION
		user.userdb.copy - The dataset name corresponding to this File. DD statements for database and native VSAM files may be omitted if the applicable dataset name information is specified in the FILE definitions in the DMCL being used by DBEZRFWD. This would not be the case if you are using the production DMCL!
Dcmsg	Idms.sysmsg.ddldcmsg	dcmsg is the DD name of the system message (DDLDCMSG) area. idms.sysmsg.ddlscmsg is the dataset name of the system message (DDLDCMSG) area.

Note: prefix refers to the high-level qualifier used when allocating the Syncsort IDMS Reorg/RSTU run-time files. Refer to the INSTDBEZ job in the appendix for further information.

Security - If security is enabled, the user submitting the DBEZRFWD job must have appropriate EXECUTE privilege for the applicable database. Additional file assignments may be needed for the user catalog and system dictionary areas depending on your security implementation. Refer to the CA-IDMS Security Administration manual for further information.

SQL - If you are reorganising or restructuring SQL areas, the catalog containing the schema definition for the areas being reorganised must be included in the DBEZRFWD job. See SYSIDMS DBNAME parameter earlier in this section for further information.

The DBEZRFWD internal jobsteps are as followed.

3.10.3 DBEZCUP1 - Process Journals

DBEZCUP1 takes as input the archive journal(s) and the "Carried-Forward" files from a previous run of this process. For the first run, the "Carried-Forward" files will be empty.

DBEZCUP1 identifies all "success units" that contain DBkeys for the areas affected by the Restructure or Unload/Reload, stripping out all run-units that are rolled-out and those that do not reference DBKeys for the affected areas.

DBEZCUP1 writes the following report to SYSLST:-

DBEZCUP1 SYSLST Report

```
 6 OCT  95  17:39        DB-EZR CATCH UP PHASE 1 REPORT            PAGE    1
           DMCL: GLB1DMCL  SUBSCHEMA: SSDBA001  SEGMENT:  DBASEG10

TOTAL RUN UNITS PROCESSED   :             98,364
UPDATE RUN UNITS PROCESSED  :              3,006
RUN UNITS CARRIED FORWARD   :                  8
```

Figure 5 DBEZCUP1 SYSLST Report

3.10.4 DBEZSUPD - Sort Run Unit data

The run unit data is sorted by ascending run unit id.

3.10.5 DBEZSJNL - Sort Journal Data

The journal data is sorted by ascending journal sequence number within run unit id.

3.10.6 DBEZCUP2 - Generate DML

This step generates the files containing all of the required DML and data necessary to replicate the updates to the affected areas against the restructured/re-sized database.

DBEZCUP2 writes the following report to SYSLST:-

DBEZCUP2 SYSLST Report

```
 6 OCT 95 17:39        DB-EZR CATCH UP PHASE 2 REPORT            PAGE    1
          DMCL: GLB1DMCL  SUBSCHEMA: SSDBA001  SEGMENT:  DBASEG10

TOTAL RUN UNITS PROCESSED  :            3,006
TOTAL DML ON JOURNAL       :           15,103
TOTAL DML GENERATED        :           19,976
```

Figure 6 DBEZCUP2 SYSLST Report

3.10.7 DBEZSDML - Sort DML File

The DML file is sorted by DML sequence number within the last journal sequence number for the DML to ensure the DMLs are applied in the order in which they completed.

3.10.8 DBEZSDBK - Sort DBZDBK3

DBZDBK3 is sorted in ascending sequence number of DBKey.

3.10.9 DBEZSXT3 - Add new DBKeys

DBZDBK2 and DBZDBK3 are merged using the sort exit DBEZSXT3 to add the new DBKeys from DBZDBK2 to
DBZDBK3 creating DBZDBK4.

3.10.10 DBEZCUP3 - Apply Updates

If you wish this step to be RESTARTABLE, you must have specified COMMIT=YES and included local mode journals in the DBEZRFWD JCL. In this case, DBEZCUP3 will issue a COMMIT for every 10,000 DML commands and will write a BFOR and AFTR checkpoint to the Syncsort IDMS Reorg/RSTU control file (DBZCNTL).

The checkpoint record will contain the last Journal and DML sequence numbers processed prior to the current and last successful COMMITs .

Refer to section 4.3.1 for further details about restart/recovery considerations following failures in this step.

DBEZCUP3 writes the following report to SYSLST:-

DBEZCUP3 SYSLST Report

```
 6 OCT 95 17:39        DB-EZR CATCH UP PHASE 3 REPORT            PAGE    1
           DMCL: GLB1DMCL   SUBSCHEMA: SSDBA001   SEGMENT:  DBASEG10

TOTAL DML PROCESSED         :            19,976
TOTAL COMMITS ISSUED        :                 1
```

Figure 7 DBEZCUP3 SYSLST Report

3.10.11 DBEZSXT4 - Sort DBKey file

The Updated DBKey file (DBZDBK3) is sorted into ascending DBKey sequence.

3.10.12 DBEZCOPY - Merge DBKey files

The sorted DBKey file (DBZDBK3) is then merged with DBZDBK2 ready for the next iteration of DBEZRFWD.

3.10.13 Backing up DBZCNTL and “Carried-Forward” files

There are three “carried-forward” files namely DBZCFRU, DBZCFJN and DBZDBK2 which are input to each successive iteration of DBEZRFWD. Apart from DBZDBK2, which is created by DBEZJOB3, these files are empty for the first execution, but for subsequent executions contain data “carried forward” from the previous execution. In the event that you need to recreate these files (for instance due to IO error), this can be done by re-submitting the current DBEZRFWD job from the DBEZCUP1 step. However, this can only be done if you have the clean “carried-forward” files from the previous iteration of DBEZRFWD. Otherwise, you would need to restart the whole “catch-up” process from the very beginning, which would be most undesirable where large amounts of data are being processed.

For this reason, it is recommended that you backup the “carried forward” files, and the DBZCNTL file before the first iteration and at the end of each successful iteration of DBEZRFWD. The method of doing this would be at your discretion. For instance, you may choose to set up a GDG, or simply to have two backup generations and alternate between them. In the event that you need to re-start an iteration of DBEZRFWD from the beginning, you can now do so by restoring the previously archived DBZCFRU, DBZCFJN, DBZDBK2 and DBZCNTL files.

If you choose to run the DBEZRFWD “catch-up” phase without local mode journals, you will also need to backup the database files at the end of each DBEZRFWD iteration (if you wish to have the possibility of restarting from the beginning of an iteration rather than going all of the way back to the beginning). If you do use local mode journals, it will not be necessary to backup the database files. Refer to the COMMIT parameter in section 3.3 and also section 4.3 for other DBEZRFWD recovery considerations.

3.11 Pre-Implementation Final Journal(s)

Once the final set of journal archive files have been processed by DBEZRFWD, the active journal data may be processed and the re-sized database brought online.

To process the active journal, proceed as follows:-

(a) Vary all affected area(s) to retrieval.

(b) Vary Journal.

(c) Feed all archive journals produced since the previous execution of DBEZRFWD, including that produced by (b) above, into the final run of DBEZRFWD.

(d) Archive both the original and the re-sized areas.

You are now ready for the implementation of the re-sized database area(s).

3.12 Implementation

There are different final implementation procedures depending on whether you want to use dynamic allocation with varying the DMCL. The different procedures are described below.

3.12.1 Restarting CV

(a) Shutdown the CV.

(b) Move the new DMCL module into the applicable run-time load library.

(c) Modify the CV start-up JCL to point to the new area(s), or rename the applicable files.

(d) Startup CV.

3.12.2 Using Dynamic Allocation

(a) Deallocate all Files for the affected area(s).

(b) Rename the old database Files.

(c) Rename the new Files to the Old names.

(d) Reallocate the Files.

(e) Move the new DMCL into the applicable run-time load library and NEWCOPY the DMCL.

Note that the dynamic allocation steps are not required if the new DSNs are included in the new DMCL.

(f) Vary affected areas back to Update.

Technical Notes

4 Technical Notes

This section contains some more detailed technical information and answers to commonly asked questions.

4.1 DBEZEXT1 - User Exit to store Direct DBKeys

If you have database records in the affected areas stored DIRECT, you may need to supply a USER EXIT as described in this section.

In actual fact. this exit is only required if DIRECT records are likely to be STORED as part of the "catch-up" process. ie. they are STORED on the production system while the UNLOAD/RELOAD is taking place. In other words, if you have a few DIRECT records that are rarely/never STORED, or not STORED during the UNLOAD/RELOAD process you would NOT need to provide this exit.

The following notes apply to the coding of such an exit.

1. The exit must be coded in assembler with an ENTRY POINT of DBEZEXT1.

The name of the module to be invoked is specified using the parameter DBEZEXT1=exitname-1 in DBEZJOB1.

DBEZRFWD step DBEZCUP3 will check for the presence of this exit if it needs to issue a STORE DML for a record with a location mode of DIRECT. If such an exit is not provided, Syncsort IDMS Reorg will be unable to store the record and the following warning message will be written to the SYSOUT file:

DB905511 DB-EZR DID NOT STORE DIRECT DBKEY 123456789.12345

2. The Exit is called with the following Register conventions.

R15=>	Entry Point Address					
R14=>	Return Address					
R13=>	Register Save Area					
R1 =>	Parameter List Pointer	=>	DBEZSSCA	DS	A	(address of SSC)
		=>	DBEZDRDA	DS	A	(address of DRD)
		=>	DBEZDBKA	DS	A	(address to return DBKey of stored record)

The DBEZEXT1 is responsible for SAVING and RESTORING registers.

3. DBEZEXT1 is responsible for dealing with the record using its own internal logic. For example, it may use the OLD DBKey to calculate a NEW DBKey and STORE the record. It may ignore the record and just generate a report, or whatever is applicable at your site.

4. In generating the internal logic, you must take into account the following points:-

(a) The RUN UNIT is already BOUND and the applicable AREAs are already OPEN in UPDATE mode.

(b) The Record BIND address is DRDDATA.

(c) The DRD record passed to the exit contains information required to deal with the current record to be stored. The DSECT layout may be found in the Appendix. Particular fields to be aware of are as follows:-

DRDDLEN	DRDDSLEN+Data Record length.
DRDDBKEY	Old DBKey of Record (as stored on production system)
DRDRID	Record Id of Record.
DRDREC	Record name.
DRDDATA	Start of Data Record

(d) After STORING the record, the exit must obtain the DBKey of the record just added to the database and move it to the field addressed by DBEZDBKA.

(e) On return from the exit, DBEZDCUP3 checks the error status in SSC. Therefore, if you want to ABORT the run, put something other than C’0000' in SSCSTAT.

4.2 DBEZEXT2 - User Exit for Called on Store or Modify of Restructured Records

May be required for Restructure. Information to be supplied.

4.3 DBEZRFWD - Restart/Recovery

This section explains what you should do in the event of a failure during the DBEZRFWD phase. Depending on the amount of update activity on the production system, and the number of off-loaded journals, the DBEZRFWD or “catch-up” phase could run for considerable time. In the event of a failure during the “catch-up” phase, it is therefore essential that the user be able to re-start from the point of failure rather than having to go back to the very beginning and start over. For this reason, Syncsort IDMS Reorg/RSTU has built-in check-point restart logic to allow straight-forward restartability.

The DBEZRFWD job consists internally of a number of steps as described in section 3.10. The internal step DBEZCUP3 is the step which issues the DML to apply the updates to the restructured/re-sized database. DBEZCUP3 is a local mode CA-IDMS job, and therefore local mode journals are required in the DBEZRFWD job to cater for potential errors in the DBEZCUP3 step. None of the other internal steps are CA-IDMS jobs. They are all essentially sort/merge/copy processes, concerned with the handling of the various work files. The recovery procedure is therefore different depending on the step which actually fails. Various messages are written to the System Log (see section 5.1) to enable you to identify the step in which the error has occurred.

You do not normally need to use the Restart or Stopafter parameters, even in the event of error recovery. This is because DBEZRFWD writes check-point information to the DBZCNTL file which it uses at the commencement of each run. You would therefore simply re-submit the DBEZRFWD job (after first correcting the problem), and DBEZRFWD will automatically restart from the beginning of the step which previously failed.

In practice, the only time that you would need to specify the RESTART EXEC parameter, would be if you needed to go back and recreate an old intermediate file, by restarting from an earlier step. For this purpose, you need to be aware of which step is responsible for the creation of which file(s). Refer to section 4.5 for a summary of the input and output files for each DBEZRFWD step.

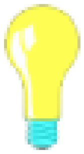
4.3.1 Recovering from errors in the DBEZCUP3 step

(A) Procedure If you are using local mode journals in DBEZRFWD

Note: If you are using local mode journals in DBEZRFWD, you must specify (or default to) COMMIT=YES in the DBEZJOB1 parameters.

In the event of an IO error, or failure of any kind in DBEZCUP3, the user must recover the re-sized database areas (using the local mode journals) before resubmitting DBEZRFWD with the local mode journals as input via the RBCKJNL DD statement.

In this case, the RESTART process will check the control file and read forward through the DML file to the last DML command successfully COMMITted to the database. Should there be a BFOR COMMIT checkpoint but no AFTR COMMIT checkpoint on the control file, the process will read forward through the local mode journal to locate the COMT record for that COMMIT. If it exists, the RESTART is from that point, otherwise from the previous COMMIT checkpoint.



It is NOT necessary in this situation to restore the DBEZCNTL and "carried-forward" files. In fact, you must NOT!

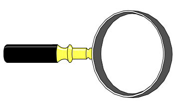
(B) Procedure If you are NOT using local mode journals in DBEZRFWD.

If you are not using local mode journals and you encounter an error in the DBEZCUP3 step, then your action is dependent on whether you are taking a backup of the database area(s) at the end of each DBEZRFWD iteration.

If you are taking backups of the database areas, then your recovery procedure is to restore the database areas AND the corresponding DBEZCNTL and "carried-forward" files from the backup taken at the end of the last successful DBEZRFWD iteration. You can then restart the "failed" iteration with the same archived journal files as input.

If you are NOT taking backups of the database areas at the end of each iteration, you will have to restart the "catch-up" phase from the beginning. That is to say, you will need to....

(i) Restore DBEZCNTL and "carried-forward" files saved prior to start of first iteration.



If you did not take back-ups at that time, you would need to initialise the "carried-forward" files by re-running DBEZJOB0, reset the DBZCNTL file by re-running DBEZJOB1 and recreate the DBZDBK2 file by re-running DBEZJOB3.

(ii) Restore database areas from the secure copy taken after the Restructure or UNLOAD/RELOAD. This assumes that you took an archive copy of the affected areas before you started the "catch-up" process. Otherwise, you really are back to square one!

(iii) Restart DBEZRFWD "catch-up" from the beginning. Ie., from the first set of archived journals.

4.3.2 Specifying the Restart Parameter

In order to permit the various internal phases (steps) to be selected and executed individually, you may specify the optional RESTART and/or STOPAFTER parameters. Where,

stepname-1 is the first step to be executed

stepname-2 is the last step to be executed.

Valid step names which may be specified for stepname-1 and stepname-2 are as follows:-

DBEZCUP1
DBEZSUPD
DBEZSJNL
DBEZCUP2

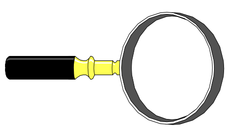
DBEZSDML
DBEZSDBK
DBEZSXT3
DBEZCUP3
DBEZSXT4
DBEZCOPY

4.3.3 Restarting an iteration of DBEZRFWD from the beginning

There may be occasions when you need to go back to the beginning of a DBEZRFWD iteration. For example, if an I/O error occurs on an intermediate file and you prefer not to have to work out which intermediate step to re-start the processing from.

If this happens during the **first iteration** of DBEZRFWD, you can accomplish this simply by restoring the DBZCNTL file and "carried-forward" files saved prior to running DBEZRFWD. If you did not backup these files at that time, you can restore them by running **DBEZJOB1** to re-initialize DBZCNTL, **DBEZJOB0** to re-initialise the carried-forward files (DBZCFRU, DBZCFJN and DBZDBK2) and then **DBEZJOB3** to recreate the DBZDB2K file. This process is also described in section 3.10.1.

If you wish to re-start from the beginning of a **subsequent iteration** of DBEZRFWD, you can only do so if you take a secure copy (backup) of the DBZCNTL file and carried-forward files (DBZDB2K, DBZCFRU and DBZCFJN) at the end of each successful iteration of DBEZRFWD. To restart an iteration, you would first need to restore the DBZCNTL file and carried-forward files from the last clean copy. Further information about backing up these files is contained in section 3.10.13.



Additional considerations apply if the current DBEZRFWD iteration has reached the DBEZCUP3 phase, that is to say, it has started to update the database. In this case, the database itself needs to be recovered. This procedure varies depending on whether you are running DBEZRFWD with local mode journals. Refer to section 4.3.1 above for details on recovering from errors in DBEZCUP3.

4.4 File Allocations

This section provides information relating to the file allocation definitions and space requirement.

Note: For the most up-to-date information refer to the INSTDBEZ install step.

FILE	BLKSIZE	LRECL	RECFM	SPACE	COMMENTS
DBZCNTL	800	80	FB	TRK,1	
RECPGXRF	2800	28	FB	CYL(1,10)	
RECPGXRS	2800	28	FB	CYL(1,10)	
UCFBTCH	6320	80	FB	TRK,1	
IDMSLDL	6320	80	FB	TRK,1	
DBZSR23	7200[1]	12	FB	TRK(1,1)	
DBZS23S	7200[1]	12	FB	TRK(1,1)	
DBZDBK2[2]	9600[1]	16	FB	CYL(1,10)	1 CYL = approx 60000 DBKeys
DBZDBK3	9600[1]	16	FB	CYL(1,10)	
DBZDBK4	9600[1]	16	FB	CYL(1,10)	
DBZDBU2	6320	10	FB	CYL(10,50)	
DBZRUPD	7384	104	FB	CYL(1,5)	
DBZRUPS	7384	104	FB	CYL(1,5)	1 CYL = approx 9000 rununits
DBZCFRU	7384	104	FB	TRK(1,1)	
DBZRTSV	6320	20	FB	TRK(1,1)	
DBZDML	1/2 track	blksize-4	VB	CYL(100,50)	Could need lots, depending on the Volumes of data processed in one go.
DBZDMLS	1/2 track	blksize-4	VB	CYL(n,n)	As DBZDML
DBZJNLU	1/2 track	blksize-4	VB	CYL(n,n)	As DBZDML

[1] Note that the Block-sizes for DBZSR23, DBZDBK2, DBZDBK3 and DBZDBK4 must all be the same.

[2] DBZDBK2 may optionally be allocated as a VSAM file. See DBEZDBK2=VSAM parameter.

DBZJNLS	1/2 track	blksize-4	VB	CYL(n,n)	As DBZDML
DBZCFJN	As SYSJRNL	As SYSJRNL	FB	CYL,(1,10)	

4.5 Summary of input and output files by job step

The following table can be used to identify the Syncsort IDMS Reorg/RSTU jobs and job-steps responsible for the creation of the different work files used during the Syncsort IDMS Reorg/RSTU process. The information may be useful in the event that you experience a problem with a particular file, and wish to restart processing from an earlier step in order to re-create a particular file. Refer to the detailed description of the particular job-steps for further information about the particular files.

JOBNAME	STEPNAME	INPUT FILES	OUTPUT FILES
DBEZJOB0	DBEZINIT		DBZCFRU DBZCFJN DBZSR23 RECPGXRF DBZDBK2
DBEZJOB1	DBEZCNTL	SYSIPT SYSIDMS	DBZCNTL RECPGXRF UCFBTCH
	SORTPGXR	RECPGXRF	RECPGXRF.SORT
	DBEZXREF	DBEZCNTL RECPGXRF.SORT SYSIDMS	RECPGXRF.SORT
DBEZJOB2	DBEZSR23	DBZCNTL Userdb (old)	DBZSR23 IDMSLDEL
	SORTSR23	DBZSR23	DBZSR23.SORT
DBEZJB3U (Unload/Reload)	DBEZSXT7	SYS005 (dbl2out)	DBEZDBU2
	DBEZSXT8	DBEZDBU2 RECPGXRF	DBEZDBK2
DBEZJB4U (Unload/Reload)	DBEZSXT7	SYS005 (dbl2out)	DBEZDBU2
	DBEZSXT8	DBEZDBU2 RECPGXRF	DBZDBK2
DBEZJB3R (DB-Reorg)	DBEZSXT6	dbreorg.dbkeys RECPGXRF	DBZDBK3 DBZSR23.SORT
	DBEZSRT6	DBEZDBK3	DBEZDBK2
DBEZJB4R (DB-Reorg only)	DBEZSXT6	dbreorg.dbkeys RECPGXRF	DBZDBK2 DBZSR23.SORT

JOBNAME	STEPNAME	INPUT FILES	OUTPUT FILES
DBEZRFWD	DBEZCUP1	DBZCNTL DBZCFRU DBZCFJN IDMSJNL	DBZCNTL DBZRUPD DBZCFRU DBZJNLU DBZRTSV
	DBEZSUPD	DBZCNTL DBZRUPD	DBZCNTL DBZRUPD.SORT
	DBEZSJNL	DBZCNTL DBZJNLU	DBZCNTL DBZJNLU.SORT
	DBEZCUP2	DBZCNTL DBZRUPD.SORT DBZJNLU.SORT DBZRTSV DBZSR23.SORT RECPGXRF	DBZCNTL DBZCFJN DBZDML DBZDBK3 DBZCFRU
	DBEZSDML	DBZCNTL DBZDML	DBZCNTL DBZDML.SORT
	DBEZSDBK	DBZCNTL DBZDBK3	DBZCNTL DBZDBK4
		DBEZSXT3	DBZCNTL
		DBZDBK2 DBZDBK4	DBZCNTL DBZDBK3
	DBEZCUP3	DBZCNTL SYSIDMS DBZDML.SORT DBZDBK3 RBCKJNL Userdb	DBZCNTL DBZDBK4 userdb SYSJRNL
	DBEZSXT4 and DBEZCOPY	DBZCNTL DBZDBK2 DBZDBK4	DBZCNTL DBZDBK2

4.6 CA-IDMS Presspack

If you have IDMS Presspack (Compression) included in the Old subschema, you must ensure that the module **IDMSUNPS** is available in the run-time library for the DBEZRFWD job. This module allows user calls to the Presspack routines, and is required by Syncsort IDMS Reorg.

4.7 COXA table name

We suggest that you use the default name for the COXA table, namely COXATAB, otherwise you will have to include the following statements in each of the EZReorg/RSTU job-streams:-

```
//COXACTL DD *
COXATAB=coxa-name
```

where coxa-name is the name you have used.

4.8 EZ-DB Dynamic Allocation

Don’t forget that you have the EZ-DB online dynamic allocation function available. This may be useful in testing EZReorg, as it allows you to dynamically deallocate and allocate files without changing the DMCL. If you have not used this facility before, simply enter the task code ALLOC8 from the DC/UCF enter next task code screen without any parameters and a HELP panel will be displayed.

For example, if you are testing by doing an UNLOAD/RELOAD without re-sizing; After completion of the “catch-up” process, just CLOSE the file, deallocate the file, re-allocate the file to the new dataset name and re-OPEN the file. You can now see the re-loaded database with any changes that were applied to the old database.

4.9 Specifying multiple PROCESS parameters

There are various considerations with regard to combining different PROCESS options in a single run.

Syncsort IDMS Reorg can carry out roll-forward operations after a database unload/reload, an index rebuild or a database restructure. This is controlled by the PROCESS= parameter

The three possible values for this parameter are REORG, INDEX or RSTU.

It is perfectly acceptable to code this parameter more than once and this is required if the roll-forward is to follow a combination of reorganisation methods. The rules for which parameters to use are:

- If an unload/reload has been carried out, there will be a file of old/new dbkeys and PROCESS=REORG is required.

- If a restructure has been done, then there will be an RSTT table and PROCESS=RSTU is required.

- When an index rebuild in done, there will be no old/new dbkeys file. In this case, code PROCESS=INDEX to tell Syncsort IDMS Reorg not to expect one. If PROCESS=REORG is coded, then PROCESS=INDEX is not required.

Valid combinations therefore are:

1. PROCESS=REORG
   PROCESS=RSTU

2. PROCESS=INDEX
   PROCESS=RSTU

All three are never required.

4.10 Old and new DMCL considerations

There are various considerations with regard to the old and new DMCL modules used.

Under certain conditions (to do with DB procs), DBEZRFWD will return IDMS status 1494 in internal step DBEZCUP2.

This error status can occur when the roll forward process encounters a record type defined with database procedures in the reload subschema. To check whether the procedures have changed from the unload subschema, a bind run unit is issued for the unload subschema in DBEZRFWD, step DBEZCUP2.

A 1494 will result if the DMCL named in SYSIDMS doesn't contain the unload segment.

In these circumstances, include both the old and new segments in the DMCL specified in the SYSIDMS for DBEZRFWD.

To simplify matters, the recommendation is to supply a DMCL that contains both unload and reload segments (where different) in all circumstances.

Appendices

5 Appendices

5.1 Syncsort IDMS Reorg/RSTU Run-time Messages

Syncsort IDMS Reorg/RSTU writes informational messages to the log as it completes each phase so that you are aware of the progress at any time. The expected messages are as follows:

```
DB905221 EZ-DB  -  DB-EZR ACTIVE DBEZCNTL
DB905521 EZ-DB  -  DB-EZR ACTIVE DBEZSR23
DB905521 EZ-DB  -  DB-EZR COMMENCING ROLLFORWARD
DB905521 DB-EZR COMPLETED DBEZCUP1
DB905521 DB-EZR COMPLETED DBEZSUPD
DB905521 DB-EZR COMPLETED DBEZSJNL
DB905521 DB-EZR COMPLETED DBEZCUP2
DB905521 DB-EZR COMPLETED DBEZSDML
DB905521 DB-EZR COMPLETED DBEZSDBK
DB905521 DB-EZR COMPLETED DBEZSXT3
DB905521 DB-EZR COMPLETED DBEZCUP3
DB905521 DB-EZR COMPLETED DBEZSXT4
DB905521 DB-EZR COMPLETED DBEZCOPY
```

5.2 #DRDDS - DML Data Record used by DBEZEXT1

```
********************************************************************
*                                                                  *
*   #DRDDS:- DB-DR DML DATA RECORD                                 *
*                                                                  *
********************************************************************
         SPACE
DRD      DSECT
DRDLEN   DS    H                   * RECORD LENGTH
         DS    H                   * FILLER
DRDLJSEQ DS    XL8                 * LAST JOURNAL SEQ NO FOR DML
DRDDSEQ  DS    F                   * DML SEQ NO
         DS    X                   * FILLER
DRDVERB  DS    X                   * VERB NO
DRDPGRUP DS    H                   * PAGE GROUP
DRDDBKEY DS    F                   * DBKEY
DRDRID   DS    H                   * RECORD ID
DRDFLG1  DS    X                   * FLAG 1
DRDIRECT EQU   X'80'               * RECORD STORED DIRECT
DRDCURCY EQU   X'40'               * MUST ESTABLISH CCY INDEX OR LDEL
DRDORDNX EQU   X'20'               * INDEX SET ORDERED NEXT
DRDORDPR EQU   X'10'               * INDEX SET ORDERED PRIOR
DRDSPAN  EQU   X'08'               * SPANNED RECORD
DRDLDEL  EQU   X'04'               * LDEL RECORD N/P IN SET
DRDSTAND EQU   X'02'               * STAND ALONE CALC OR DIRECT
DRDFLG2  DS    X                   * FLAG 1
DRDCSR7  EQU   X'80'               * SET OWNER SYSTEM INDEX CURRENT
DRDCLAS  EQU   X'40'               * SET LAST SYSTEM INDEX CURRENT
DRDCFIR  EQU   X'20'               * SET FIRST SYSTEM INDEX CURRENT
DRDPOWN  EQU   X'10'               * NEXT/PRIOR POINTS TO OWNER
DRDREC   DS    CL18                * RECORD NAME
DRDSET   DS    0CL18               * SET NAME
DRDDATA  DS    0H                  * RECORD DATA
DRDDSLEN EQU   *-DRD               * DSECT LENGTH
         ORG   DRDDATA+L'DRDSET
DRDDSETL EQU   *-DRD               * DSECT LENGTH WITH SET NAME
```

5.3 MVS JCL

In this section can be found listings of the source members contained within the EZ-DB installation source library. The JCL must be modified as indicated with reference to the applicable section in this user guide.

5.3.1 INSTDBEZ JCL

The following member (**INSTDBEZ**) is included on the installation tape source library containing the JCL necessary to allocate all of the work-files used by the EZReorg/RSTU process.

```
//*********************************************************************
//*    I N T E G R A T E D   I N S T A L L A T I O N   S T E P S   *
//*********************************************************************
//*
//*  INSTDBEZ - Integrated Installation JCL for Syncsort IDMS Reorg
//*
//*  This member contains the default installation steps required to
//*  complete the ez-db integrated install for Syncsort IDMS Reorg.
//*
//*  Please see the Syncsort IDMS Reorg User Guide for recommendations
//*  for space allocations.
//*
//*  Please note that various file DCB details have changed in
//*  R6.0 from those used in R5.x.
//*
//*********************************************************************
//*
//*  INSTALLATION JOB CONTROL LANGUAGE.
//*
//*  THE JOB STREAM CONSISTS OF THE FOLLOWING STEPS:
//*
//*  1.    ALLOCDF - ALLOCATE FILES REQUIRED FOR Syncsort IDMS Reorg
//*
//*********************************************************************
//*
//*  The following changes should be made before submitting
//*  this job stream. It is suggested that you save a copy of this
//*  file first.
//*
//*  1.  CHANGE prefix TO THE PREFIX REQUIRED FOR THE DSNs OF ALL
//*      Syncsort IDMS Reorg RUN TIME DATASETS.
//*
//*
//*  2.  ALLOCDF:  Override the default space allocations as required.
//*                Block sizes provided are not necessarily the most
//*                efficient for large volumes of data. They should
//*                adjusted to be half track, or near half track
//*                if RECFM=FB when it must also be a multiple of
//*                LRECL.
//*
//*                Note that the DCB attributes for DBEZCFJN must be
//*                the same as your journal archive files. ie the
//*                same as SYSJRNL in the DMCL used by IDMSAJNL.
//*                This may vary by DMCL.
//*
//*                DBEZRFWD requires local mode journalling unless
//*                COMMIT=NO was specified to DBEZCNTL. We
//*                recommend that you set up a GDG with the same
//*                attributes as the GDG of the journal archive
//*                files input to this process. The sample JCL
//*                supplied for DBEZRFWD assumes a GDG is in place.
//*
//*  3.  DBEZDBK2: If you intend to use a VSAM file for DBZDBK2 then
//*                use the DD DUMMY version of this DD statement.
//*                By default this is commented out.
```

```
//*********************************************************************
//*************************************************************
//*                                                           *
//*   ALLOCATE Syncsort IDMS Reorg RUNTIME DATASETS           *
//*                                                           *
//*************************************************************
//*
//ALLOCDF  EXEC PGM=IEFBR14,COND=(0,LT)
//DBZCNTL   DD DSN=prefix.DBEZCNTL,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=800,LRECL=80),
//             UNIT=SYSDA,SPACE=(TRK,1)
//RECPGXRF  DD DSN=prefix.RECPGXRF,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=2800,LRECL=28),
//             UNIT=SYSDA,SPACE=(CYL,(1,10))
//RECPGXRS  DD DSN=prefix.RECPGXRF.SORT,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=2800,LRECL=28),
//             UNIT=SYSDA,SPACE=(CYL,(1,10))
//UCFBTCH   DD DSN=prefix.UCFBATCH,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=6320,LRECL=80),
//             UNIT=SYSDA,SPACE=(TRK,1)
//IDMSLDL   DD DSN=prefix.IDMSLDEL,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=6320,LRECL=80),
//             UNIT=SYSDA,SPACE=(TRK,1)
//DBZSR23   DD DSN=prefix.DBEZSR23,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=7200,LRECL=12),
//             UNIT=SYSDA,SPACE=(TRK,(1,1))
//DBZS23S   DD DSN=prefix.DBEZSR23.SORT,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=7200,LRECL=12),
//             UNIT=SYSDA,SPACE=(TRK,(1,1))
//*DBZDBK2  DD DUMMY,DCB=(RECFM=FB,DSORG=PS,BLKSIZE=9600,LRECL=16),
//DBZDBK2   DD DSN=prefix.DBEZDBK2,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=9600,LRECL=16),
//             UNIT=SYSDA,SPACE=(CYL,(1,10))
//DBZDBK3   DD DSN=prefix.DBEZDBK3,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=9600,LRECL=16),
//             UNIT=SYSDA,SPACE=(CYL,(1,10))
//DBZDBK4   DD DSN=prefix.DBEZDBK4,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=9600,LRECL=16),
//             UNIT=SYSDA,SPACE=(CYL,(1,10))
//DBZDBU2   DD DSN=prefix.DBEZDBU2,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=6320,LRECL=10),
//             UNIT=SYSDA,SPACE=(CYL,(10,50),RLSE)
//DBZRUPD   DD DSN=prefix.DBEZRUPD,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=7384,LRECL=104),
//             UNIT=SYSDA,SPACE=(CYL,(1,5))
//DBZRUPS   DD DSN=prefix.DBEZRUPD.SORT,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=7384,LRECL=104),
//             UNIT=SYSDA,SPACE=(CYL,(1,5))
//DBZCFRU   DD DSN=prefix.DBEZCFRU,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=7384,LRECL=104),
//             UNIT=SYSDA,SPACE=(TRK,(1,1))
//DBZRTSV   DD DSN=prefix.DBEZRTSV,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=6320,LRECL=20),
//             UNIT=SYSDA,SPACE=(TRK,(1,1))
//DBZDML    DD DSN=prefix.DBEZDML,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=VB,DSORG=PS,BLKSIZE=23476,LRECL=23472),
//             UNIT=SYSDA,SPACE=(CYL,(100,50))
//DBZDMLS   DD DSN=prefix.DBEZDML.SORT,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=VB,DSORG=PS,BLKSIZE=23476,LRECL=23472),
//             UNIT=SYSDA,SPACE=(CYL,(100,50))
//DBZJNLU   DD DSN=prefix.DBEZJNLU,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=VB,DSORG=PS,BLKSIZE=23476,LRECL=23472),
//             UNIT=SYSDA,SPACE=(CYL,(100,50))
//DBZJNLS   DD DSN=prefix.DBEZJNLU.SORT,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=VB,DSORG=PS,BLKSIZE=23476,LRECL=23472),
//             UNIT=SYSDA,SPACE=(CYL,(100,50))
//DBZCFJN   DD DSN=prefix.DBEZCFJN,DISP=(,CATLG,DELETE),
//             DCB=(RECFM=FB,DSORG=PS,BLKSIZE=nnnnn,LRECL=nnnnn),
//             UNIT=SYSDA,SPACE=(CYL,(1,10))
```

```
//*  The DCB attributes for DBEZCFJN must be the same as SYSJNRL
//*  in the DMCL used by IDMSAJNL.
//*
//*  Uncomment the following DD card if using the optional 'Merged
//*  Journal' option for Data Sharing environments
//*
//*DBZMUTL DD DSN=prefix.DBEZMUTL,DISP=(,CATLG,DELETE),
//*           DCB=(RECFM=FB,DSORG=PS,BLKSIZE=800,LRECL=80),
//*           UNIT=SYSDA,SPACE=(TRK,1)
//**********************************************************************
//* E N D   O F   I N T E G R A T E D   I N S T A L L
//**********************************************************************
```

5.3.2 DBEZJOB0 JCL

```
//*****************************************************************
//*        DBEZJOB0 - Initialize EZReorg Files                   *
//*                   If you intend to use a VSAM file for       *
//*                   DBZDBK2 then use the DD DUMMY version      *
//*                   of it's DD statement on the first step and *
//*                   change the next step to COND=(0,LT).       *
//*****************************************************************
//DBEZINIT EXEC PGM=DBEZINIT,REGION=9M
//STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSLST   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DBZCFRU  DD DSN=prefix.DBEZCFRU,DISP=SHR
//DBZCFJN  DD DSN=prefix.DBEZCFJN,DISP=SHR
//DBZSR23  DD DSN=prefix.DBEZSR23.SORT,DISP=SHR
//RECPGXRF DD DSN=prefix.RECPGXRF,DISP=SHR
//DBZDBK2  DD DSN=prefix.DBEZDBK2,DISP=SHR
//*DBZDBK2  DD DUMMY,DCB=BLKSIZE=20
//*****************************************************************
//*                                                              *
//*   ALLOCATE A DBEZDBK2 VSAM FILE                              *
//*                                                              *
//*****************************************************************
//DBK2ALOC EXEC PGM=IDCAMS,REGION=1536K,COND=(0,LE)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DELETE prefix.DBEZDBK2
  DEFINE CLUSTER( -
                NAME (prefix.DBEZDBK2)           -
                VOL (volser)                     -
                CYLINDERS ( n n)                 -
                INDEXED                          -
                REUSE                            -
                SHAREOPTIONS (2 3)               -
                )                                -
           DATA ( -
                NAME (prefix.DBEZDBK2.DATA)      -
                RECORDSIZE(16 16)                -
                KEYS(6,2)                        -
                SHAREOPTIONS (2 3)               -
                )                                -
           INDEX ( -
                NAME (prefix.DBEZDBK2.INDEX)     -
                SHAREOPTIONS (2 3)               -
                IMBED                            -
                )
/*
//*
```

Refer to section 3.2 for further details.

5.3.3 DBEZJOB1 JCL

```
//*****************************************************************
//*          DBEZJOB1 - CREATE SYNCSORT IDMS REORG CONTROL FILE    *
//*****************************************************************
//DBEZCNTL EXEC PGM=DBEZCNTL,REGION=9M
//STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//         DD DSN=CAIDMS.OLD.DBALOAD,DISP=SHR
//         DD DSN=CAIDMS.LOADLIB,DISP=SHR
//SYSLST   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DBZCNTL  DD DSN=prefix.DBEZCNTL,DISP=SHR
//UCFBTCH  DD DSN=prefix.UCFBATCH,DISP=SHR
//RECPGXRF DD DSN=prefix.RECPGXRF,DISP=SHR
//SYSIDMS  DD *
DMCL=dmcl-name
DBNAME=olddbnam       * Specify old dbname when using multi-seg Reorg
/*
//SYSIPT   DD *
sysipt parameters
/*
//*
//*****************************************************************
//*          SYNCSORT IDMS REORG SORT RECID/PG XREF FILE           *
//*****************************************************************
//SORTPGXR EXEC PGM=ICEMAN,REGION=9076K
//SORTIN   DD DSN=prefix.RECPGXRF,DISP=SHR
//SORTOUT  DD DSN=prefix.RECPGXRF.SORT,DISP=SHR
//SORTWK01 DD DSN=&&SRTWK1,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK02 DD DSN=&&SRTWK2,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK03 DD DSN=&&SRTWK3,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK04 DD DSN=&&SRTWK4,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK05 DD DSN=&&SRTWK5,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK06 DD DSN=&&SRTWK6,SPACE=(CYL,5),UNIT=SYSDA
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 RECORD TYPE=F
 SORT FIELDS=(1,4,BI,A)
//*
//*****************************************************************
//*          UPDATE RECPGXRF FILE WITH NEW SEGMENT DETAILS         *
//*****************************************************************
//DBEZXREF EXEC PGM=DBEZXREF,REGION=32M,COND=(0,LT)
//STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//         DD DSN=CAIDMS.NEW.DBALOAD,DISP=SHR
//         DD DSN=CAIDMS.LOADLIB,DISP=SHR
//SYSLST   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DBZCNTL  DD DSN=prefix.DBEZCNTL,DISP=SHR
//RECPGXRF DD DSN=prefix.RECPGXRF.SORT,DISP=SHR
//SYSIDMS  DD *
DMCL=dmcl-name
DBNAME=newdbnam       * Specify new dbname when using multi-seg Reorg
/*
//*
//*  Uncomment the following step if using the optional 'Merged
//*  Journal' option for Data Sharing environments
//*
//*DBEZMUIN EXEC PGM=DBEZMUIN,REGION=32M,COND=(0,LT)
//*STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//*SYSLST   DD SYSOUT=*
//*SYSUDUMP DD SYSOUT=*
//*DBZMUTL  DD DSN=prefix.DBEZMUTL,DISP=SHR
//*DBZCNTL  DD DSN=prefix.DBEZCNTL,DISP=SHR
//*
```

Refer to Section 3.3 For further information.

5.3.4 DBEZJOB2 JCL

```
//*****************************************************************
//*        DBEZJOB2 - SYNCSORT IDMS REORG CREATE SR2/SR3 DATA    *
//*****************************************************************
//DBEZSR23 EXEC PGM=DBEZSR23,REGION=9M
//STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//         DD DSN=CAIDMS.OLD.DBALOAD,DISP=SHR
//SYSLST   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DBZCNTL  DD DSN=prefix.DBEZCNTL,DISP=SHR
//DBZSR23  DD DSN=prefix.DBEZSR23,DISP=SHR
//IDMSLDL  DD DSN=prefix.IDMSLDEL,DISP=SHR
//*
//userdb  DD DSN=user.userdb.copy,DISP=SHR
//*
//*****************************************************************
//*        SYNCSORT IDMS REORG SORT SR2/SR3 DBKEYS                           *
//*****************************************************************
//SORTSR23 EXEC PGM=ICEMAN,REGION=9076K
//SORTIN   DD DSN=prefix.DBEZSR23,DISP=SHR
//SORTOUT  DD DSN=prefix.DBEZSR23.SORT,DISP=SHR
//SORTWK01 DD DSN=&&SRTWK1,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK02 DD DSN=&&SRTWK2,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK03 DD DSN=&&SRTWK3,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK04 DD DSN=&&SRTWK4,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK05 DD DSN=&&SRTWK5,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK06 DD DSN=&&SRTWK6,SPACE=(CYL,5),UNIT=SYSDA
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 RECORD TYPE=F
 SORT FIELDS=(3,2,BI,A,9,4,BI,A)
//*
```

Refer to Section 3.5 for further information.

5.3.5 DBEZJB3R JCL

```
//*****************************************************************
//*     DBEZJB3R                                                  *
//*     Use:  DB Keys output from DB-Reorg - non-VSAM DB2K file  *
//*                                                               *
//*     Note: This job is run when running DB-Reorg and using    *
//*           the default NON-VSAM DBEZDBK2 file.                 *
//*                                                               *
//*           If you use a VSAM file for DBEZDBK2 then you must  *
//*           use job DBEZJB4R.                                   *
//*                                                               *
//*           If you are running UNLOAD/RELOAD then use job      *
//*           DBEZJB3U or DBEZJB4U                                *
//*                                                               *
//*****************************************************************
//*     GENERATE FILE OF OLD AND NEW DBKEYS USING OUTPUT         *
//*     FROM DB-REORG.                                            *
//*****************************************************************
//DBEZSXT6 EXEC PGM=ICEMAN,REGION=9076K,COND=(0,LE)
//SORTIN   DD DSN=EZ-DB.DBREORG.DBKEYS.file-1,DISP=SHR
//         DD DSN=EZ-DB.DBREORG.DBKEYS.file-n,DISP=SHR
//SORTOUT  DD DUMMY,DCB=*.SORTIN
//DBZDBK2  DD DSN=prefix.DBEZDBK3,DISP=SHR
//DBZSR23  DD DSN=prefix.DBEZSR23.SORT,DISP=SHR
//RECPGXRF DD DSN=prefix.RECPGXRF.SORT,DISP=SHR
//SORTWK01 DD DSN=&&SRTWK1,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK02 DD DSN=&&SRTWK2,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK03 DD DSN=&&SRTWK3,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK04 DD DSN=&&SRTWK4,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK05 DD DSN=&&SRTWK5,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK06 DD DSN=&&SRTWK6,SPACE=(CYL,5),UNIT=SYSDA
//DBEZLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSABEND DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 RECORD TYPE=F,LENGTH=16
 SORT FIELDS=(1,4,BI,A)
 MODS E35=(DBEZSXT6,2500,DBEZLIB,N)
//*
//*****************************************************************
//*        SYNCSORT IDMS REORG SORT DBK2 ON PAGE GROUP AND DBKEY          *
//*****************************************************************
//DBEZSRT6 EXEC PGM=ICEMAN,REGION=9076K
//SORTIN   DD DSN=prefix.DBEZDBK3,DISP=SHR
//SORTOUT  DD DSN=prefix.DBEZDBK2,DISP=SHR
//SORTWK01 DD DSN=&&SRTWK1,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK02 DD DSN=&&SRTWK2,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK03 DD DSN=&&SRTWK3,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK04 DD DSN=&&SRTWK4,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK05 DD DSN=&&SRTWK5,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK06 DD DSN=&&SRTWK6,SPACE=(CYL,5),UNIT=SYSDA
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 RECORD TYPE=F
 SORT FIELDS=(3,6,BI,A)
//*
```

Refer to Section 3.8.2 for further information.

5.3.6 DBEZJB3U JCL

```
//*****************************************************************
//*     DBEZJB3U                                                  *
//*     Use:  DB Keys output from Unload/Reload - non-VSAM DB2K  *
//*                                                               *
//*     Note: This job is run when running Unload/Reload and     *
//*           using the default NON-VSAM DBEZDBK2 file.          *
//*                                                               *
//*           If you use a VSAM file for DBEZDBK2 then you must  *
//*           use job DBEZJB4U.                                   *
//*                                                               *
//*           If you are running DB-Reorg then use job           *
//*           DBEZJB3U or DBEZJB4U                               *
//*                                                               *
//*****************************************************************
//*     GENERATE FILE OF OLD AND NEW DBKEYS USING OUTPUT         *
//*     FROM IDMSDBL2.                                            *
//*****************************************************************
//DBEZSXT7 EXEC PGM=DBEZSXT7,REGION=8M
//STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSABEND DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DBL2OUT  DD DSN=EZ-DB.DBL2OUT,DISP=SHR,DCB=BUFNO=32
//DBZDBK2  DD DSN=prefix.DBEZDBU2,DISP=SHR,DCB=BUFNO=32
//*
//DBEZSXT8 EXEC PGM=ICEMAN,REGION=8M
//SORTIN   DD DSN=prefix.DBEZDBU2,DISP=SHR,DCB=BUFNO=32
//SORTOUT  DD DUMMY,DCB=*.SORTIN
//DBZDBK2  DD DSN=prefix.DBEZDBK2,DISP=SHR,DCB=BUFNO=32
//RECPGXRF DD DSN=prefix.RECPGXRF.SORT,DISP=SHR
//SORTWK01 DD DSN=&&SRTWK1,SPACE=(CYL,50),UNIT=SYSDA
//SORTWK02 DD DSN=&&SRTWK2,SPACE=(CYL,50),UNIT=SYSDA,SEP=SORTWK01
//SORTWK03 DD DSN=&&SRTWK3,SPACE=(CYL,50),UNIT=SYSDA,
//            SEP=(SORTWK01,SORTWK02)
//DBEZLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSABEND DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 RECORD TYPE=F
 SORT FIELDS=(7,4,BI,A)
 MODS E35=(DBEZSXT8,2500,DBEZLIB,N)
//*
```

Refer to Section 3.8.1 for further information.

5.3.7 DBEZJB4R JCL

```
//*****************************************************************
//*     DBEZJB4R                                                  *
//*     Use:  DB Keys output from DB-Reorg - VSAM DB2K file       *
//*                                                               *
//*     Note: This job is run when running DB-Reorg and using    *
//*           a VSAM DBEZDBK2 file.                               *
//*                                                               *
//*           If you use a non-VSAM file for DBEZDBK2 then you   *
//*           must use job DBEZJB3R.                              *
//*                                                               *
//*           If you are running UNLOAD/RELOAD then use job       *
//*           DBEZJB3U or DBEZJB4U                                *
//*                                                               *
//*****************************************************************
//*     GENERATE FILE OF OLD AND NEW DBKEYS USING OUTPUT          *
//*     FROM DB-REORG.                                            *
//*****************************************************************
//DBEZSXT6 EXEC PGM=ICEMAN,REGION=9076K,COND=(0,LE)
//SORTIN   DD DSN=EZ-DB.DBREORG.DBKEYS.file-1,DISP=SHR
//         DD DSN=EZ-DB.DBREORG.DBKEYS.file-n,DISP=SHR
//SORTOUT  DD DUMMY,DCB=*.SORTIN
//DBZDBK2  DD DSN=prefix.DBEZDBK2,DISP=SHR
//DBZSR23  DD DSN=prefix.DBEZSR23.SORT,DISP=SHR
//RECPGXRF DD DSN=prefix.RECPGXRF.SORT,DISP=SHR
//SORTWK01 DD DSN=&&SRTWK1,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK02 DD DSN=&&SRTWK2,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK03 DD DSN=&&SRTWK3,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK04 DD DSN=&&SRTWK4,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK05 DD DSN=&&SRTWK5,SPACE=(CYL,5),UNIT=SYSDA
//SORTWK06 DD DSN=&&SRTWK6,SPACE=(CYL,5),UNIT=SYSDA
//DBEZLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSABEND DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 RECORD TYPE=F,LENGTH=16
 SORT FIELDS=(1,4,BI,A)
 MODS E35=(DBEZVXT6,2500,DBEZLIB,N)
//*
```

Refer to Section 3.9.2 for further information.

5.3.8 DBEZJB4U JCL

```
//*****************************************************************
//*     DBEZJB4U                                                  *
//*     Use:  DB Keys output from Unload/Reload - VSAM DB2K       *
//*                                                               *
//*     Note: This job is run when running Unload/Reload and      *
//*           using a VSAM DBEZDBK2 file.                         *
//*                                                               *
//*           If you use a non-VSAM file for DBEZDBK2 then you    *
//*           must use job DBEZJB3U.                              *
//*                                                               *
//*           If you are running DB-Reorg then use job            *
//*           DBEZJB3U or DBEZJB4U                                *
//*                                                               *
//*****************************************************************
//*     GENERATE FILE OF OLD AND NEW DBKEYS USING OUTPUT          *
//*     FROM IDMSDBL2.                                            *
//*****************************************************************
//DBEZSXT7 EXEC PGM=DBEZSXT7,REGION=8M
//STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSABEND DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DBL2OUT  DD DSN=EZ-DB.DBL2OUT,DISP=SHR,DCB=BUFNO=32
//DBZDBK2  DD DSN=prefix.DBEZDBU2,DISP=SHR,DCB=BUFNO=32
//*
//DBEZSXT8 EXEC PGM=ICEMAN,REGION=8M
//SORTIN   DD DSN=prefix.DBEZDBU2,DISP=SHR,DCB=BUFNO=32
//SORTOUT  DD DUMMY,DCB=*.SORTIN
//DBZDBK2  DD DSN=prefix.DBEZDBK2,DISP=SHR,DCB=BUFNO=32
//RECPGXRF DD DSN=prefix.RECPGXRF.SORT,DISP=SHR
//SORTWK01 DD DSN=&&SRTWK1,SPACE=(CYL,50),UNIT=SYSDA
//SORTWK02 DD DSN=&&SRTWK2,SPACE=(CYL,50),UNIT=SYSDA,SEP=SORTWK01
//SORTWK03 DD DSN=&&SRTWK3,SPACE=(CYL,50),UNIT=SYSDA,
//            SEP=(SORTWK01,SORTWK02)
//DBEZLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSABEND DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
 RECORD TYPE=F
 SORT FIELDS=(7,4,BI,A)
 MODS E35=(DBEZVXT8,2500,DBEZLIB,N)
//*
```

Refer to Section 3.9.1 for further information.

5.3.9 DBEZRFWD JCL

```
//*****************************************************************
//*        DBEZRFWD - SYNCSORT IDMS REORG ONE STEP ROLL FORWARD  *
//*        NOTES:  1. It is recommended that the following files *
//*                   be backed up at the end of this job step.  *
//*                   prefix.DBEZCNTL                            *
//*                   prefix.DBEZCFRU                            *
//*                   prefix.DBEZCFJN                            *
//*                   prefix.DBEZDBK2                            *
//*                2. If COMMIT=NO was specified in the SYSIPT   *
//*                   parameters input to DBEZCNTL then the      *
//*                   local mode journal SYSJRNL is not required *
//*                   and may be dummied out. In this case we    *
//*                   recommend the IDMS database files be       *
//*                   backed up prior to each iteration of       *
//*                   DBEZRFWD.                                  *
//*                3. caidms.old.dbaload is only required if     *
//*                   the old and new subschemas have different  *
//*                   names and reside in different load libs.   *
//*****************************************************************
//DBEZRFWD EXEC PGM=DBEZRFWD,REGION=9M
//STEPLIB  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//         DD DSN=CAIDMS.NEW.DBALOAD,DISP=SHR
//         DD DSN=CAIDMS.OLD.DBALOAD,DISP=SHR      * OPTIONAL
//         DD DSN=CAIDMS.LOADLIB,DISP=SHR
//DBZCNTL  DD  DSN=prefix.DBEZCNTL,DISP=SHR
//DBZCFJN  DD  DSN=prefix.DBEZCFJN,DISP=SHR
//DBZCFRU  DD  DSN=prefix.DBEZCFRU,DISP=SHR
//DBZRTSV  DD  DSN=prefix.DBEZRTSV,DISP=SHR
//DBZRUPD  DD  DSN=prefix.DBEZRUPD,DISP=SHR
//DBZRUPS  DD  DSN=prefix.DBEZRUPD.SORT,DISP=SHR
//DBZDML   DD  DSN=prefix.DBEZDML,DISP=SHR
//DBZDMLS  DD  DSN=prefix.DBEZDML.SORT,DISP=SHR
//DBZSR23  DD  DSN=prefix.DBEZSR23.SORT,DISP=SHR
//DBZDBK2  DD  DSN=prefix.DBEZDBK2,DISP=SHR
//DBZDBK3  DD  DSN=prefix.DBEZDBK3,DISP=SHR
//DBZDBK4  DD  DSN=prefix.DBEZDBK4,DISP=SHR
//DBZJNLU  DD  DSN=prefix.DBEZJNLU,DISP=SHR,DCB=BUFNO=32
//DBZJNLS  DD  DSN=prefix.DBEZJNLU.SORT,DISP=SHR,DCB=BUFNO=32
//RECPGXRF DD  DSN=prefix.RECPGXRF.SORT,DISP=SHR
//IDMSJNL  DD  DSN=idms.journal.file-1,DISP=SHR,DCB=BUFNO=32
//         DD  DSN=idms.journal.file-n,DISP=SHR
//RBCKJNL  DD  DUMMY
//SYSJRNL  DD DSN=prefix.SYSJRNL(+1),DISP=(,CATLG,DELETE),
//            DCB=(ajnl.model,RECFM=FB,LRECL=lrecl,DSORG=PS),
//            UNIT=unit
//SXT3IN01 DD DSN=prefix.DBEZDBK2,DISP=SHR
//SXT3IN02 DD DSN=prefix.DBEZDBK4,DISP=SHR
//SXT4IN01 DD DSN=prefix.DBEZDBK3,DISP=SHR
//SXT4IN02 DD DSN=prefix.DBEZDBK2,DISP=SHR
//SORTWK01 DD DSN=&&SRTWK1,SPACE=(CYL,500),UNIT=SYSDA
//SORTWK02 DD DSN=&&SRTWK2,SPACE=(CYL,500),UNIT=SYSDA
//SORTWK03 DD DSN=&&SRTWK3,SPACE=(CYL,500),UNIT=SYSDA
//SXT3WK01 DD DSN=&&SRTWK31,SPACE=(CYL,10),UNIT=SYSDA
//SXT3WK02 DD DSN=&&SRTWK32,SPACE=(CYL,10),UNIT=SYSDA
//SXT4WK01 DD DSN=&&SRTWK41,SPACE=(CYL,10),UNIT=SYSDA
//SXT4WK02 DD DSN=&&SRTWK42,SPACE=(CYL,10),UNIT=SYSDA
//SYSLST   DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIDMS  DD *
dmcl=dmcl-name
dbname=new-dbname       * Required when using multiple segs with REORG
//*
//userdb   DD DSN=user.userdb.copy,DISP=SHR
//*
//*  Uncomment the following DD card if using the optional 'Merged
```

```
//*  Journal' option for Data Sharing environments
//*
//*DBZMUTL  DD DSN=prefix.DBEZMUTL,DISP=SHR
```

Refer to Section 3.10 For further information.

5.4 Data Sharing – Merged Journal Option

For sites that have implemented Data Sharing across multiple CVs, Syncsort IDMS Reorg can still be used but requires the separately licensed 'Merged Journal' option.

To implement the 'Merged Journal Option', make the following JCL changes:

1. In the INSTDBEZ job, uncomment the allocation of the DBZMUTL DD card.

2. In DBEZJOB1 job, uncomment the DBEZMUIN step.

3. In the DBEZRFWD job, uncomment the DBZMUTL DD card.

Syncsort IDMS Reorg is then run as normal, but with the journal data from the output of the 'MERGE ARCHIVE' utility. Refer to the CA IDMS UTILITIES Guide for details of the 'MERGE ARCHIVE' option