MAXM Reorg Solutions for IMS User Guide

Supporting

Version 4.8 of CHANGE RECORDING FACILITY
Version 4.8 of FAST REORG FACILITY/EP
Version 4.8 of LOADPLUS/EP for IMS
Version 4.8 of MAXM Reorg/EP for IMS
Version 4.8 of MAXM Reorg/EP Express for IMS
Version 4.8 of MAXM Reorg/EP with Online/Defrag Feature for IMS
Version 4.8 of MAXM Reorg/Online for IMS
Version 4.8 of SECONDARY INDEX UTILITY/EP
Version 4.8 of UNLOAD PLUS/EP for IMS

December 2011
Contacting BMC Software

You can access the BMC Software website at http://www.bmc.com. From this website, you can obtain information about the company, its products, corporate offices, special events, and career opportunities.

**United States and Canada**

<table>
<thead>
<tr>
<th>Address</th>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
</table>
| BMC SOFTWARE INC  
2101 CITYWEST BLVD  
HOUSTON TX 77042-2827  
USA | 1 713 918 8800 or 1 800 841 2031 | 1 713 918 8000 |

**Outside United States and Canada**

<table>
<thead>
<tr>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>+01 713 918 8800</td>
<td>+01 713 918 8000</td>
</tr>
</tbody>
</table>
Customer support

You can obtain technical support by using the BMC Support Central website or by contacting Customer Support by telephone or e-mail. To expedite your inquiry, see “Before contacting BMC.”

Support website

You can obtain technical support from BMC 24 hours a day, 7 days a week at http://www.bmc.com/support. From this website, you can

- read overviews about support services and programs that BMC offers
- find the most current information about BMC products
- search a database for issues similar to yours and possible solutions
- order or download product documentation
- download products and maintenance
- report an issue or ask a question
- subscribe to receive proactive e-mail alerts when new product notices are released
- find worldwide BMC support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

Support by telephone or e-mail

In the United States and Canada, if you need technical support and do not have access to the web, call 1 800 537 1813 or send an e-mail message to customer_support@bmc.com. (In the subject line, enter SupID:<yourSupportContractID>, such as SupID:12345). Outside the United States and Canada, contact your local support center for assistance.

Before contacting BMC

Have the following information available so that Customer Support can begin working on your issue immediately:

- product information
  - product name
  - product version (release number)
  - license number and password (trial or permanent)

- operating system and environment information
  - machine type
  - operating system type, version, and service pack or other maintenance level such as PUT or PTF
  - system hardware configuration
  - serial numbers
  - related software (database, application, and communication) including type, version, and service pack or maintenance level

- sequence of events leading to the issue

- commands and options that you used

- messages received (and the time and date that you received them)
  - product error messages
  - messages from the operating system, such as file system full
  - messages from related software
License key and password information

If you have questions about your license key or password, use one of the following methods to get assistance:

- Send an e-mail message to customer_support@bmc.com.
# Contents

## About this book

Summary of changes .......................................................... 32

### Chapter 1  Introduction ................................................. 35

Overview ............................................................................. 36  
  MAXM Reorg/Online solution ........................................... 37  
  MAXM Reorg/EP solutions .............................................. 39  
  MAXM Reorg/EP Express solution .................................... 40  
  Comparison of utilities .................................................. 40  
  Getting started .............................................................. 42  

Goals of data management .................................................. 42  
  Improve data availability ............................................. 42  
  Conserve data resources ............................................. 44  
  Preserve data integrity ................................................ 46  
  Use familiar structure ................................................ 47  

MAXM Reorg/Online features and benefits ......................... 48  

Reorg function features and benefits .................................. 49  
  Data set name swapping ............................................. 51  
  Logically related databases ........................................ 52  

Unload function features and benefits ................................ 52  

Reload function features and benefits ................................. 54  

Index Build function features and benefits ....................... 55  

Prefix Resolution and Update function features and benefits 58  

Online/Defrag function features and benefits ..................... 60  

DBA Toolkit features and benefits ...................................... 61  
  Automated data collection ......................................... 61  
  Automatic analysis .................................................. 62  
  Automatic configuration ........................................... 62  
  Conditional reorganization ....................................... 62  
  Control block disassembly ........................................ 62  
  DBD mapping .......................................................... 63  
  Exception management ............................................. 63  
  JCL generator ......................................................... 63  
  Repository management .......................................... 63  

Database Management Console ........................................... 64  
  Statistical information ............................................... 64  

MAXM Reorg/Online processing phases ............................... 64  
  Phase 1: Set up the database for reorganization ............... 67  
  Phase 2: Reorganize the database .................................. 68
### Phase 3: Prefix resolution and update. 
- Page: 69

### Phase 4: Roll forward the changes. 
- Page: 69

### Phase 5: Make the database available. 
- Page: 69

### Phase 6: Correct the logs. 
- Page: 70

## System requirements
- Page: 70

## Limitations
- Page: 72

## Chapter 2 Reorg function (Fast Reorg Facility/EP)  
- Overview: Page 80
- Requirements: Page 82
- JCL requirements: Page 83
  - Standard Reorg function reorganization: Page 83
  - Reorganization using an HD Unload file as input: Page 84
  - Reorganization for logically related databases: Page 86
  - Reorganization to an image copy: Page 91
  - Reorganizing multiple databases: Page 92
- DD statements: Page 92
- PLUSIN examples: Page 93
- JCL considerations: Page 94
- Unused files: Page 97
- Keywords: Page 97
- Global options module: Page 101
- Reports: Page 102
- Post-processing phase: Page 103
- Empty database initialization: Page 103
- Operational considerations: Page 104
  - DBRC authorization: Page 104
  - Secondary indexes: Page 105
  - Image copies: Page 106
  - Database replication: Page 108

## Chapter 3 Online Reorg function  
- Overview: Page 109
- Requirements: Page 113
- Handling BMPs with AR/CTL: Page 114
- JCL requirements: Page 115
- DD statements: Page 116
- Keywords: Page 119
- PLUSIN examples: Page 123
- Global options module: Page 124
- Report: Page 124
- Unload phase: Page 125
- Post-processing phase: Page 125
- Automated operator interface: Page 126
  - BMPs: Page 127
  - DBR commands: Page 128
<table>
<thead>
<tr>
<th>Chapter 4</th>
<th>Online/Defrag feature</th>
<th>135</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td></td>
<td>135</td>
</tr>
<tr>
<td>Record mode process</td>
<td></td>
<td>137</td>
</tr>
<tr>
<td>Scan mode process</td>
<td></td>
<td>139</td>
</tr>
<tr>
<td>Block Mode Process</td>
<td></td>
<td>139</td>
</tr>
<tr>
<td>Online/Defrag options</td>
<td></td>
<td>140</td>
</tr>
<tr>
<td>Online/Defrag Record Analysis</td>
<td></td>
<td>144</td>
</tr>
<tr>
<td>Extractor and Recommendation Selector</td>
<td></td>
<td>146</td>
</tr>
<tr>
<td>Analyzer</td>
<td></td>
<td>150</td>
</tr>
<tr>
<td>Online reorganizations</td>
<td></td>
<td>153</td>
</tr>
<tr>
<td>System flow</td>
<td></td>
<td>153</td>
</tr>
<tr>
<td>JCL requirements</td>
<td></td>
<td>154</td>
</tr>
<tr>
<td>Examples</td>
<td></td>
<td>156</td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td>159</td>
</tr>
<tr>
<td>Reports</td>
<td></td>
<td>159</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 5</th>
<th>Multiple databases</th>
<th>165</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td></td>
<td>165</td>
</tr>
<tr>
<td>Requirements</td>
<td></td>
<td>167</td>
</tr>
<tr>
<td>Commands and keywords</td>
<td></td>
<td>168</td>
</tr>
<tr>
<td>JCL</td>
<td></td>
<td>169</td>
</tr>
<tr>
<td>Reports</td>
<td></td>
<td>170</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 6</th>
<th>Unload function</th>
<th>173</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td></td>
<td>174</td>
</tr>
<tr>
<td>JCL requirements</td>
<td></td>
<td>175</td>
</tr>
<tr>
<td>Non-parallel unload</td>
<td></td>
<td>175</td>
</tr>
<tr>
<td>Parallel unload</td>
<td></td>
<td>176</td>
</tr>
<tr>
<td>DD statements</td>
<td></td>
<td>176</td>
</tr>
<tr>
<td>PLUSIN examples</td>
<td></td>
<td>177</td>
</tr>
<tr>
<td>Snapshot Copy function</td>
<td></td>
<td>178</td>
</tr>
<tr>
<td>Keywords</td>
<td></td>
<td>178</td>
</tr>
<tr>
<td>Global options modules</td>
<td></td>
<td>181</td>
</tr>
<tr>
<td>Create using ISPF interface</td>
<td></td>
<td>182</td>
</tr>
<tr>
<td>Create using batch job</td>
<td></td>
<td>184</td>
</tr>
<tr>
<td>Reports</td>
<td></td>
<td>184</td>
</tr>
<tr>
<td>HD Unload data set differences</td>
<td></td>
<td>185</td>
</tr>
<tr>
<td>Performance considerations</td>
<td></td>
<td>186</td>
</tr>
<tr>
<td>Buffers</td>
<td></td>
<td>186</td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td>187</td>
</tr>
</tbody>
</table>

DBD changes during reorganization ................................................. 129
DELTA IMS DB/DC installed ......................................................... 130
DELTA IMS DB/DC not installed .................................................. 132
Database recovery ................................................................. 133
Non-disruptive maintenance ...................................................... 133
### Chapter 8  Index Build function

Overview ........................................... 244
JCL requirements ................................... 245
  Build secondary indexes only ....................... 245
  Build secondary indexes and primary index in one job step ....... 246
  Build primary index only ........................... 247
DD statements ......................................... 247
PLUSIN examples .................................... 248
Sort considerations .................................. 249
Keywords ............................................. 249
Global options modules .............................. 251
  Create using ISPF interface ........................ 252
  Create using batch job .............................. 253
Reports ............................................... 253
Simplified index creation and maintenance .............. 254
Other functions .................................... 257
  Load indexes from an existing DFSURWF1 file .......... 257
  Build indexes by scanning the primary database ....... 257
  Build primary indexes .............................. 258
HALDBs ............................................... 259
Create a new secondary index ......................... 260
Rebuild an existing secondary index .................... 261
Maintain a secondary index .......................... 261
Create an empty secondary index ...................... 262

### Chapter 9  Copy function

Overview ........................................... 263
Healing EPS pointers in a HALDB .................... 268
  POINTER CHECKER PLUS ......................... 268
  Healing EPS pointers with the Copy function ....... 269

### Chapter 10  Working with logical relationships

Overview ........................................... 272
High-performance Prefix Resolution function ........... 272
  JCL requirements ................................... 275
Concurrent Prefix Update function .................... 277
  JCL requirements ................................... 278
Commands and keywords ................................ 278
Global options modules ................................ 280
  Create using ISPF interface ........................ 281
  Create using batch job .............................. 283
Reports ............................................... 284
Performance considerations .......................... 285
Other prefix resolution and update functions .......... 286
  IMS-compatible Prefix Resolution function .......... 286
  Two-step Prefix Resolution function .............. 289
  Stand-alone Prefix Update function ............... 295
<table>
<thead>
<tr>
<th>Secondary Index Resolution function</th>
<th>297</th>
</tr>
</thead>
<tbody>
<tr>
<td>JCL requirements</td>
<td>299</td>
</tr>
<tr>
<td>Work File 1 Generator</td>
<td>300</td>
</tr>
<tr>
<td>JCL requirements</td>
<td>302</td>
</tr>
<tr>
<td>Work File Splitter</td>
<td>304</td>
</tr>
<tr>
<td>JCL requirements</td>
<td>305</td>
</tr>
</tbody>
</table>

**Chapter 11  Image Copy function**

<table>
<thead>
<tr>
<th>Overview</th>
<th>307</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Copy function with the Online Reorg function</td>
<td>308</td>
</tr>
<tr>
<td>Considerations and limitations</td>
<td>309</td>
</tr>
<tr>
<td>JCL requirements</td>
<td>310</td>
</tr>
<tr>
<td>ICPSYSIN statements</td>
<td>311</td>
</tr>
<tr>
<td>ICPSYSIN commands</td>
<td>312</td>
</tr>
<tr>
<td>ICPSYSIN keywords</td>
<td>313</td>
</tr>
<tr>
<td>User-controlled options</td>
<td>317</td>
</tr>
<tr>
<td>Global options modules</td>
<td>318</td>
</tr>
<tr>
<td>Database-specific options members</td>
<td>319</td>
</tr>
<tr>
<td>Control statement keywords</td>
<td>319</td>
</tr>
<tr>
<td>Image Copy function global options modules</td>
<td>319</td>
</tr>
<tr>
<td>Database-specific options modules</td>
<td>324</td>
</tr>
<tr>
<td>Reports</td>
<td>325</td>
</tr>
<tr>
<td>Image copy data sets</td>
<td>325</td>
</tr>
<tr>
<td>Explicit allocation of image copy data sets</td>
<td>325</td>
</tr>
<tr>
<td>Dynamic allocation of image copy data sets</td>
<td>326</td>
</tr>
<tr>
<td>Virtual image copies</td>
<td>330</td>
</tr>
<tr>
<td>Output image copy write options</td>
<td>331</td>
</tr>
<tr>
<td>Access method control</td>
<td>331</td>
</tr>
<tr>
<td>Buffer pool size specification</td>
<td>332</td>
</tr>
<tr>
<td>Image copy block size settings</td>
<td>333</td>
</tr>
<tr>
<td>POINTER CHECKER PLUS interface</td>
<td>334</td>
</tr>
<tr>
<td>Monitoring and reporting options</td>
<td>334</td>
</tr>
<tr>
<td>Monitor the progress of the job</td>
<td>335</td>
</tr>
<tr>
<td>Free space analysis information</td>
<td>335</td>
</tr>
<tr>
<td>PDX history limit settings</td>
<td>335</td>
</tr>
<tr>
<td>GROUP control statement</td>
<td>335</td>
</tr>
<tr>
<td>User-defined GROUP processing</td>
<td>336</td>
</tr>
<tr>
<td>DBRC-defined GROUP processing</td>
<td>337</td>
</tr>
</tbody>
</table>

**Chapter 12  RECON Substitution Facility**

<table>
<thead>
<tr>
<th>Overview</th>
<th>339</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSF Extract function</td>
<td>340</td>
</tr>
<tr>
<td>RSF Extract keywords</td>
<td>341</td>
</tr>
<tr>
<td>RSF execution intercept</td>
<td>342</td>
</tr>
<tr>
<td>Creating a test database</td>
<td>343</td>
</tr>
</tbody>
</table>
## Chapter 13 | JCL requirements

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>346</td>
</tr>
<tr>
<td>EXEC statement</td>
<td>347</td>
</tr>
<tr>
<td>DD statements</td>
<td>349</td>
</tr>
<tr>
<td>AMSOUT DD</td>
<td>351</td>
</tr>
<tr>
<td>AMSPDS DD</td>
<td>352</td>
</tr>
<tr>
<td>ANLPKEYO DD</td>
<td>353</td>
</tr>
<tr>
<td>ANLPRINT DD</td>
<td>353</td>
</tr>
<tr>
<td>BMCMWG DD</td>
<td>353</td>
</tr>
<tr>
<td>BMCPRTN DD</td>
<td>354</td>
</tr>
<tr>
<td>BMASETUP DD</td>
<td>355</td>
</tr>
<tr>
<td>BMCTRACE DD</td>
<td>355</td>
</tr>
<tr>
<td>CRFRDER/CRFRDER2 DD</td>
<td>356</td>
</tr>
<tr>
<td>DFSRESLB DD</td>
<td>357</td>
</tr>
<tr>
<td>DFSUINnn DD</td>
<td>358</td>
</tr>
<tr>
<td>DFSUINPT DD</td>
<td>358</td>
</tr>
<tr>
<td>DFSURCDS DD</td>
<td>359</td>
</tr>
<tr>
<td>DFSURGU1 DD</td>
<td>360</td>
</tr>
<tr>
<td>DFSURGU2 DD</td>
<td>360</td>
</tr>
<tr>
<td>DFSURGUnn DD</td>
<td>361</td>
</tr>
<tr>
<td>DFSURGmnn DD</td>
<td>362</td>
</tr>
<tr>
<td>DFSURIDX DD</td>
<td>363</td>
</tr>
<tr>
<td>DFSURWF1 DD</td>
<td>364</td>
</tr>
<tr>
<td>DFSURWF2 DD</td>
<td>365</td>
</tr>
<tr>
<td>DFSURWF3 DD</td>
<td>366</td>
</tr>
<tr>
<td>DMNCNTLI DD</td>
<td>367</td>
</tr>
<tr>
<td>EXTRACTO DD</td>
<td>367</td>
</tr>
<tr>
<td>HLUDURIDX DD</td>
<td>367</td>
</tr>
<tr>
<td>HSRCNTRLI DD</td>
<td>368</td>
</tr>
<tr>
<td>ICPPRINT DD</td>
<td>368</td>
</tr>
<tr>
<td>IEFRDER/IEFRDER2 DD</td>
<td>369</td>
</tr>
<tr>
<td>IMS DD</td>
<td>370</td>
</tr>
<tr>
<td>OLRKEYS DD</td>
<td>370</td>
</tr>
<tr>
<td>PCPDANIP DD</td>
<td>371</td>
</tr>
<tr>
<td>PDX DD</td>
<td>371</td>
</tr>
<tr>
<td>PLUSIN DD</td>
<td>372</td>
</tr>
<tr>
<td>PLUSLIST DD</td>
<td>373</td>
</tr>
<tr>
<td>PLUSOUT DD</td>
<td>373</td>
</tr>
<tr>
<td>print DD</td>
<td>373</td>
</tr>
<tr>
<td>PRPURIDX DD</td>
<td>375</td>
</tr>
<tr>
<td>PRPURWF1 DD</td>
<td>375</td>
</tr>
<tr>
<td>RECONx DD</td>
<td>376</td>
</tr>
<tr>
<td>SEQERRORR DD</td>
<td>377</td>
</tr>
<tr>
<td>SORTIN DD</td>
<td>377</td>
</tr>
<tr>
<td>SORTWIK DD</td>
<td>378</td>
</tr>
<tr>
<td>STEPLIB DD</td>
<td>378</td>
</tr>
<tr>
<td>SYSCOUT DD</td>
<td>379</td>
</tr>
<tr>
<td>SYSUDUMP DD</td>
<td>380</td>
</tr>
</tbody>
</table>
Chapter 14  User-controlled options  381

Overview  .................................................. 384
How the utilities build options  ................................... 385
Setting options. .................................................. 387
   Online Reorg function ...................................... 388
   Reorg function ............................................. 388
   Individual MAXM Reorg functions  ......................... 388
Commands and keywords overview.  ......................... 388
   PLUSIN control statement syntax  ......................... 404
Keyword descriptions .......................................... 406
   ABNDNO ................................................. 406
   AGN .................................................... 407
   ALLOWLC .............................................. 407
   ALLOWLP .............................................. 407
   AMSOUT ............................................... 408
   AOI ..................................................... 409
   AOIUSER ............................................... 410
   AUTHERR ............................................... 410
   AUTOSTOP ............................................. 411
   BIPF .................................................. 413
   BUFSIZE .............................................. 413
   CKUPDATE ............................................ 414
   COMPRESS, EXPAND ................................... 415
   CREORG ............................................. 419
   DBD .................................................. 420
   DBDSGRP ............................................. 420
   DBR .................................................. 421
   DBRC (Index Build and Reorg functions) .................. 422
   DBRC (Prefix Resolution/Update function) ................ 423
   DDN .................................................. 423
   DESC (Image Copy function)  .............................. 424
   DFSDBUX1 ........................................... 424
   DFSURG1 ............................................. 425
   DYNALLOC .......................................... 430
   EOJACCES ........................................... 433
   FBFF .................................................. 434
   FFTDHAL ............................................. 434
   FILSZ ............................................... 437
   FIXED ............................................... 437
   FMTALL .............................................. 438
   FMTRA A .............................................. 439
   FORCERP ............................................. 439
   FREQ .................................................. 440
   FSPF .................................................. 441
   FUNC ............................................... 442
   GSGNAME ............................................. 442
   HDAMFSPF .......................................... 443
   HDSORT .............................................. 443
   HIPO .................................................. 446
<table>
<thead>
<tr>
<th>Variable</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>HIST</td>
<td>447</td>
</tr>
<tr>
<td>HSAMOUT</td>
<td>448</td>
</tr>
<tr>
<td>IC</td>
<td>449</td>
</tr>
<tr>
<td>ICACC</td>
<td>449</td>
</tr>
<tr>
<td>ICALLOC</td>
<td>450</td>
</tr>
<tr>
<td>ICAVGREC</td>
<td>451</td>
</tr>
<tr>
<td>ICBUF</td>
<td>451</td>
</tr>
<tr>
<td>ICCATF</td>
<td>452</td>
</tr>
<tr>
<td>ICDATAACL</td>
<td>453</td>
</tr>
<tr>
<td>ICDCSB</td>
<td>454</td>
</tr>
<tr>
<td>ICEXPDT</td>
<td>454</td>
</tr>
<tr>
<td>ICEDG</td>
<td>455</td>
</tr>
<tr>
<td>ICLIKE</td>
<td>455</td>
</tr>
<tr>
<td>ICMGMTCL</td>
<td>456</td>
</tr>
<tr>
<td>ICNEEDED</td>
<td>457</td>
</tr>
<tr>
<td>ICP</td>
<td>458</td>
</tr>
<tr>
<td>ICPREF</td>
<td>459</td>
</tr>
<tr>
<td>ICRETPD</td>
<td>460</td>
</tr>
<tr>
<td>ICSPACE</td>
<td>461</td>
</tr>
<tr>
<td>ICSTORCL</td>
<td>462</td>
</tr>
<tr>
<td>ICSUNIT</td>
<td>463</td>
</tr>
<tr>
<td>ICVOLCNT</td>
<td>463</td>
</tr>
<tr>
<td>ICVOLSER</td>
<td>464</td>
</tr>
<tr>
<td>IDCAMS</td>
<td>464</td>
</tr>
<tr>
<td>IILDS</td>
<td>468</td>
</tr>
<tr>
<td>IMSID</td>
<td>470</td>
</tr>
<tr>
<td>INDD</td>
<td>471</td>
</tr>
<tr>
<td>INDDX</td>
<td>472</td>
</tr>
<tr>
<td>INPUT</td>
<td>473</td>
</tr>
<tr>
<td>INST</td>
<td>476</td>
</tr>
<tr>
<td>IRLMGRP</td>
<td>477</td>
</tr>
<tr>
<td>KEYS</td>
<td>478</td>
</tr>
<tr>
<td>LIM</td>
<td>478</td>
</tr>
<tr>
<td>LIMITS</td>
<td>479</td>
</tr>
<tr>
<td>LOGDSN</td>
<td>483</td>
</tr>
<tr>
<td>LOGEXPDT, LOGRETPD</td>
<td>485</td>
</tr>
<tr>
<td>LOGSMS</td>
<td>485</td>
</tr>
<tr>
<td>LOGUNIT</td>
<td>486</td>
</tr>
<tr>
<td>LOGSPACE</td>
<td>486</td>
</tr>
<tr>
<td>LPRCK</td>
<td>487</td>
</tr>
<tr>
<td>MBI</td>
<td>488</td>
</tr>
<tr>
<td>MINREC</td>
<td>489</td>
</tr>
<tr>
<td>MODE</td>
<td>489</td>
</tr>
<tr>
<td>MONITOR, MONUSERS, ROUTCODE, DESCODE</td>
<td>490</td>
</tr>
<tr>
<td>NAME</td>
<td>493</td>
</tr>
<tr>
<td>NEWDBD</td>
<td>494</td>
</tr>
<tr>
<td>NOAUTH</td>
<td>494</td>
</tr>
<tr>
<td>NOTIFY</td>
<td>495</td>
</tr>
<tr>
<td>OLDDBD</td>
<td>496</td>
</tr>
</tbody>
</table>
OSAMINIT .............................................................. 499
OSAMMAX .............................................................. 500
OUTDD ................................................................. 501
OUTDDX ................................................................. 501
OVFLOWONLY ......................................................... 503
PADCHAR .............................................................. 503
PARALLEL ............................................................. 504
PARMBLK .............................................................. 507
PART (database partitioning) ...................................... 508
PCP ...................................................................... 517
PDX (Unload, Reload, and Index Build functions) .......... 518
PDX (Image Copy function) ....................................... 518
PDX (Prefix Resolution/Update function) ................. 519
PDX (Online/Defrag function) .................................. 519
PDXPARMS ............................................................ 520
PRIMIND ............................................................... 520
PRTOLC ................................................................. 521
PRTOLP ................................................................. 521
xxxPRINT .............................................................. 522
PSB ...................................................................... 523
PTRCHECK ............................................................ 524
PTRERROR ............................................................. 524
REORGMSG ............................................................ 525
REPORTS .............................................................. 526
RESUME ............................................................... 528
RMBYTES .............................................................. 529
ROUTCDE (Image Copy function) .............................. 530
SCAN—space search method for HIDAM and PHIDAM .... 531
SCAN—space search method for HDAM and PHDAM ...... 531
SECINDEX ............................................................. 534
SEGPOS ................................................................. 535
SEGPOSO ............................................................... 536
SEGSEL ................................................................. 536
SEQERROR ............................................................. 537
SHARE ................................................................. 540
SIUDSN ................................................................. 541
SIUSORT ............................................................... 542
SNAPSHOT .............................................................. 543
SORTIDX ............................................................... 544
SORTIN ................................................................. 544
SORTWORK ........................................................... 545
SPILL .................................................................... 547
SSID ...................................................................... 547
STATS ................................................................. 548
SWAP ................................................................. 548
SYNCTAPE ............................................................ 554
TIME ................................................................. 556
TRUNC ................................................................. 556
UCBITMAP ............................................................. 557
### Chapter 15 Reports

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>580</td>
</tr>
<tr>
<td>Report destinations (most functions)</td>
<td>580</td>
</tr>
<tr>
<td>Report destinations (Prefix Resolution/Update function)</td>
<td>581</td>
</tr>
<tr>
<td>Report destinations (Image Copy function)</td>
<td>581</td>
</tr>
<tr>
<td>Reports stored in the PDX</td>
<td>581</td>
</tr>
<tr>
<td>Summary of reports</td>
<td>582</td>
</tr>
<tr>
<td>ISPF interface to view reports</td>
<td>584</td>
</tr>
<tr>
<td>Limit the number of report sets saved</td>
<td>584</td>
</tr>
<tr>
<td>Bad Block/Pointer report</td>
<td>585</td>
</tr>
<tr>
<td>Concurrent Reorganization</td>
<td>586</td>
</tr>
<tr>
<td>Database Summary</td>
<td>586</td>
</tr>
<tr>
<td>Data Set Buffer Statistics report</td>
<td>587</td>
</tr>
<tr>
<td>Data Set Group Summary</td>
<td>588</td>
</tr>
<tr>
<td>Dataset List</td>
<td>591</td>
</tr>
<tr>
<td>Dataset Statistics</td>
<td>591</td>
</tr>
<tr>
<td>Datasets Used</td>
<td>592</td>
</tr>
<tr>
<td>DBD Change Summary</td>
<td>592</td>
</tr>
<tr>
<td>DBD Summary</td>
<td>592</td>
</tr>
<tr>
<td>Disassembled DBD</td>
<td>594</td>
</tr>
<tr>
<td>Distribution of Database Record Sizes</td>
<td>594</td>
</tr>
<tr>
<td>Distribution of Free Space</td>
<td>595</td>
</tr>
<tr>
<td>Distribution of HDAM RAP Chain Lengths</td>
<td>595</td>
</tr>
<tr>
<td>Global Specifications</td>
<td>596</td>
</tr>
<tr>
<td>HDAM Root Addressable Area Statistics</td>
<td>597</td>
</tr>
<tr>
<td>HDAM Root Placement Relative to Its Home Block</td>
<td>598</td>
</tr>
<tr>
<td>HD Sort - Record Counts report</td>
<td>599</td>
</tr>
<tr>
<td>IDCAMS System Services</td>
<td>600</td>
</tr>
<tr>
<td>Index Load Output Results</td>
<td>600</td>
</tr>
<tr>
<td>Largest Database Records</td>
<td>600</td>
</tr>
<tr>
<td>Listing of PLUSIN Control Statements</td>
<td>601</td>
</tr>
</tbody>
</table>
Logical Relationship Information ........................................ 601
Logical Relationship Summary ........................................ 601
Message Log ........................................................................ 604
Options in Effect................................................................. 604
Output Data Set Summary .................................................. 605
Output Image Copy Summary .............................................. 605
Parent/Child Segment Statistics ......................................... 606
Partition Stats ................................................................. 606
Process Summary ............................................................. 607
Processing Log ..................................................................... 607
Processing Summary .......................................................... 608
Reports for DBIL Type Reorganization ................................ 608
Reports for DBR Type Reorganization ................................ 609
Secondary Index Definition ............................................... 609
Secondary Index Relationship Summary ............................. 611
Segment I/O Requirements Matrix ....................................... 612
Segment Statistics ............................................................ 612
Status of Segments Read from HD Unload Dataset .............. 613
Twin Segment Statistics ..................................................... 614
User Exit Routine Action .................................................... 614
Variable Length Segment Statistics ...................................... 615
VL Segment Split Statistics ................................................ 616

Appendix A Diagnostic procedures .................................... 617

Return codes ...................................................................... 617
  Online Reorg function and Reorg function ......................... 618
  Reload function ............................................................... 618
  Index Build function ........................................................ 619
  Prefix Resolution/Update function .................................... 619
Diagnostic DD statements .................................................. 620
  $BMCS$MOD DD ........................................................... 620
  PLUSDUMP DD ............................................................. 620
  PLUSLIST DD .............................................................. 620
  PLUSOUT DD ............................................................... 621
  SYSUDUMP DD ............................................................ 621
Trace ................................................................................. 621
  BMCTRACE DD ............................................................ 621
  APITRACE DD .............................................................. 622
  TRACE keyword ........................................................... 622
Online Reorg function problem resolution ............................ 622
Using the Trace facility with the Reorg function .................. 623
  Keywords ...................................................................... 623
  Coding procedures ........................................................ 624
Figures

MAXM Reorg/Online overview ................................................................. 38
MAXM Reorg/Online reorganization .................................................... 66
Reorganization with Unload function with Snapshot Copy ...................... 67
Reorg function system flow ................................................................. 80
Inputs and outputs of standard Reorg function reorganization ................. 81
Reorg function JCL ................................................................................ 83
Reorg function JCL when using HD Unload file as input ............................ 84
Logical relationships .............................................................................. 87
Reorg function JCL for single-step reorganization .................................... 88
Reorg function for JCL for multi-step reorganizations ............................... 89
Reorg function JCL for SWAP(x,IC) ........................................................ 91
MAXM Reorg/Online system flow .......................................................... 112
Online Reorg function JCL .................................................................... 115
WFI BMP processing ............................................................................. 128
JCL for OLDDBD with DELTA IMS DB/DC ............................................ 131
Record 1 before record mode reorganization .......................................... 138
Record 1 after record mode reorganization ............................................. 138
Online/Defrag option construction ......................................................... 141
Online/Defrag Record Analysis system flow .......................................... 144
Online/Defrag Extractor and Recommendation Selector JCL .................. 146
Online/Defrag Analyzer JCL ................................................................. 150
Online reorganization system flow ......................................................... 153
Minimum JCL requirements for Online/Defrag function .......................... 156
JCL for Online Reorganization using record mode ................................... 157
JCL for Online Reorganization using scan mode ..................................... 158
JCL for Online Reorganization block mode ............................................ 159
Multiple database reorganization task flow ............................................ 166
REORG PLUSIN example for multiple databases .................................... 169
COPY PLUSIN example for multiple databases ...................................... 169
Reorg function JCL for multiple database reorganization ....................... 170
Sample output for multiple database reorganization .............................. 171
BMCMSG output for Copy function ..................................................... 172
Unload function system flow ............................................................... 174
Unload function JCL (non-parallel mode) .............................................. 175
Unload function JCL (parallel mode) ..................................................... 176
Unload function ISPF interface panel map ............................................. 183
System flow of the DB Scan function .................................................. 189
Sample JCL to run the DB Scan function ............................................ 189
Sample JCL for Database Selection facility ......................................... 192
Database Selection facility sample output (printed format) .................... 197
Tables

DBA Toolkit features .................................................. 61
MAXM Reorg/Online phases ........................................... 64
System requirements .................................................... 70
dnames and data set names ........................................... 81
Reorg function keywords ............................................. 97
Global options module keywords ................................. 101
Reorg function reports ............................................... 102
Valid data set name combinations for authorization .......... 104
Online Reorg function keywords ................................. 119
Online Reorg function report ..................................... 124
Control statements for JCL for OLDDBD with DELTA IMS DB/DC ... 131
Online/Defrag commands ............................................. 141
Online/Defrag options ............................................... 141
Components and processes of the Online/Defrag Record Analysis .......... 145
Online/Defrag Extractor and Recommendation Selector options ........ 148
Database Analysis Reorganization Recommendation report fields .... 149
Analyzer recommendation report fields ........................ 152
Components of the online reorganization ...................... 154
Online/Defrag reports ................................................. 160
Options in Effect report fields .................................... 161
DBD Summary report fields ....................................... 162
Reorganization Results report fields .............................. 163
Commands for multiple database reorganizations and COPY ........ 168
Keyword changes for multiple database reorganizations and copies .... 168
Unload function keywords ......................................... 179
Ways to specify options for the Unload function .............. 181
Unload function reports ............................................. 184
Unload function options that affect performance .............. 187
DBSELECT command keywords .................................... 193
USERREAD command keywords .................................. 204
TRACE command keywords ....................................... 206
Reload function keywords ......................................... 216
Ways to specify options for Reload function .................. 219
Reload function reports ............................................. 222
Reload function options that affect performance .............. 225
PLUSIN DD keywords ............................................... 232
USERLOAD command keywords ................................ 238
Reload function API DL/I status codes .......................... 239
Index Build function keywords .................................. 249
Ways to specify options for the Index Build function .......... 251
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSPF keyword</td>
<td>441</td>
</tr>
<tr>
<td>Image copy group function keyword</td>
<td>442</td>
</tr>
<tr>
<td>CSG name keyword</td>
<td>442</td>
</tr>
<tr>
<td>Free space percentage factor keyword</td>
<td>443</td>
</tr>
<tr>
<td>Invoke HD Sort keyword</td>
<td>443</td>
</tr>
<tr>
<td>HDSORT keyword restrictions</td>
<td>446</td>
</tr>
<tr>
<td>High performance option</td>
<td>446</td>
</tr>
<tr>
<td>HIPO subkeywords</td>
<td>447</td>
</tr>
<tr>
<td>Image copy history generations to retain keyword</td>
<td>447</td>
</tr>
<tr>
<td>Use of the HSAMOUT keyword</td>
<td>448</td>
</tr>
<tr>
<td>Output image copy ddname keyword</td>
<td>449</td>
</tr>
<tr>
<td>Output image copy access method keyword</td>
<td>449</td>
</tr>
<tr>
<td>Dynamically allocate output image copy keyword</td>
<td>450</td>
</tr>
<tr>
<td>Output image copy SMS AVGREC keyword</td>
<td>451</td>
</tr>
<tr>
<td>Output image copy buffer algorithm keyword</td>
<td>451</td>
</tr>
<tr>
<td>ICBUF values</td>
<td>452</td>
</tr>
<tr>
<td>Catalog output image copies keyword</td>
<td>453</td>
</tr>
<tr>
<td>Output image copy SMS DATACLAS keyword</td>
<td>453</td>
</tr>
<tr>
<td>Output image copy model DSCB keyword</td>
<td>454</td>
</tr>
<tr>
<td>Output image copy expiration date keyword</td>
<td>454</td>
</tr>
<tr>
<td>Create output image copy as GDG keyword</td>
<td>455</td>
</tr>
<tr>
<td>Create output image copy LIKE model keyword</td>
<td>456</td>
</tr>
<tr>
<td>Output image copy SMS MGMTCLAS keyword</td>
<td>456</td>
</tr>
<tr>
<td>ICNEEDED flag keyword</td>
<td>457</td>
</tr>
<tr>
<td>ICNEEDED keyword and the ICNEEDED flag</td>
<td>458</td>
</tr>
<tr>
<td>Image Copy function keyword</td>
<td>458</td>
</tr>
<tr>
<td>Output image copy data set name prefix keyword</td>
<td>460</td>
</tr>
<tr>
<td>Output image copy retention period keyword</td>
<td>460</td>
</tr>
<tr>
<td>Output image copy space allocation keyword</td>
<td>461</td>
</tr>
<tr>
<td>ICSPACE values</td>
<td>462</td>
</tr>
<tr>
<td>Output image copy SMS STORCLAS keyword</td>
<td>462</td>
</tr>
<tr>
<td>Output image copy unit keyword</td>
<td>463</td>
</tr>
<tr>
<td>Output image copy volume count keyword</td>
<td>463</td>
</tr>
<tr>
<td>Output image copy volume serial keyword</td>
<td>464</td>
</tr>
<tr>
<td>IDCAMS keyword</td>
<td>464</td>
</tr>
<tr>
<td>ILDS keyword and the BUILD command</td>
<td>468</td>
</tr>
<tr>
<td>ILDS keyword and the RELOAD and REORG commands</td>
<td>469</td>
</tr>
<tr>
<td>IMSID keyword</td>
<td>471</td>
</tr>
<tr>
<td>Input data set keyword</td>
<td>471</td>
</tr>
<tr>
<td>Read secondary index records from non-DFSURWF1 file</td>
<td>472</td>
</tr>
<tr>
<td>Input unload file keyword</td>
<td>473</td>
</tr>
<tr>
<td>INPUT(UNLDFILE) keyword and other keywords</td>
<td>475</td>
</tr>
<tr>
<td>Image copy installation name keyword</td>
<td>477</td>
</tr>
<tr>
<td>IRLM group keyword</td>
<td>477</td>
</tr>
<tr>
<td>KEYS keyword</td>
<td>478</td>
</tr>
<tr>
<td>Limit between orphans keyword</td>
<td>479</td>
</tr>
<tr>
<td>Parallel unload limits keyword</td>
<td>479</td>
</tr>
<tr>
<td>LIMITS keyword and other keywords</td>
<td>480</td>
</tr>
<tr>
<td>Log data set name keyword</td>
<td>483</td>
</tr>
<tr>
<td>Keyword</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>LOGDSN masks</td>
<td>484</td>
</tr>
<tr>
<td>Log expiration date and retention period keywords</td>
<td>485</td>
</tr>
<tr>
<td>LOGSMS keywords</td>
<td>486</td>
</tr>
<tr>
<td>LOGUNIT keyword</td>
<td>486</td>
</tr>
<tr>
<td>Log space keyword</td>
<td>487</td>
</tr>
<tr>
<td>Bypass retrieval of LPCKs keyword</td>
<td>487</td>
</tr>
<tr>
<td>LPCK keyword and other keywords</td>
<td>488</td>
</tr>
<tr>
<td>MBI keyword</td>
<td>488</td>
</tr>
<tr>
<td>MINREC keyword</td>
<td>489</td>
</tr>
<tr>
<td>MODE keyword</td>
<td>489</td>
</tr>
<tr>
<td>Monitoring keywords</td>
<td>490</td>
</tr>
<tr>
<td>Interval values</td>
<td>493</td>
</tr>
<tr>
<td>User ID values</td>
<td>493</td>
</tr>
<tr>
<td>Image copy group name keyword</td>
<td>493</td>
</tr>
<tr>
<td>Database restructure keyword (UNLOAD command)</td>
<td>494</td>
</tr>
<tr>
<td>DBRC authorization keyword</td>
<td>495</td>
</tr>
<tr>
<td>Notification keyword</td>
<td>495</td>
</tr>
<tr>
<td>Database restructure keyword (REORG command)</td>
<td>496</td>
</tr>
<tr>
<td>OSAMINIT database keyword</td>
<td>500</td>
</tr>
<tr>
<td>8-GB OSAM database keyword</td>
<td>500</td>
</tr>
<tr>
<td>OUTDDD keyword</td>
<td>501</td>
</tr>
<tr>
<td>Write secondary index records to non-DFSURWF1 file keyword</td>
<td>502</td>
</tr>
<tr>
<td>Overflow only keyword</td>
<td>503</td>
</tr>
<tr>
<td>PADCHAR keyword</td>
<td>503</td>
</tr>
<tr>
<td>Parallel processing keyword</td>
<td>504</td>
</tr>
<tr>
<td>Image Copy function global options module name keyword</td>
<td>507</td>
</tr>
<tr>
<td>Partitioning keyword</td>
<td>508</td>
</tr>
<tr>
<td>LIMITS and PART keyword combinations</td>
<td>511</td>
</tr>
<tr>
<td>HALDB partitioning examples</td>
<td>513</td>
</tr>
<tr>
<td>IMS/ESA partition support product examples</td>
<td>514</td>
</tr>
<tr>
<td>Invoke POINTER CHECKER PLUS keyword</td>
<td>517</td>
</tr>
<tr>
<td>PDX keyword (Unload, Reload, and Index Build functions)</td>
<td>518</td>
</tr>
<tr>
<td>Image Copy function PDX data set name keyword</td>
<td>518</td>
</tr>
<tr>
<td>PDX keyword (Prefix Resolution/Update function)</td>
<td>519</td>
</tr>
<tr>
<td>PDX keyword</td>
<td>519</td>
</tr>
<tr>
<td>PDX member name keyword</td>
<td>520</td>
</tr>
<tr>
<td>Primary index keyword</td>
<td>520</td>
</tr>
<tr>
<td>PRIMIND keyword and other keywords</td>
<td>520</td>
</tr>
<tr>
<td>Print logical child orphans keyword</td>
<td>521</td>
</tr>
<tr>
<td>Print logical parent orphans keyword</td>
<td>522</td>
</tr>
<tr>
<td>Reports data set keyword</td>
<td>522</td>
</tr>
<tr>
<td>PSB keyword</td>
<td>523</td>
</tr>
<tr>
<td>Check physical pointers keyword</td>
<td>524</td>
</tr>
<tr>
<td>Pointer error keyword</td>
<td>524</td>
</tr>
<tr>
<td>REORGMESG keyword</td>
<td>525</td>
</tr>
<tr>
<td>Reports keyword</td>
<td>526</td>
</tr>
<tr>
<td>REPORTS subkeywords</td>
<td>526</td>
</tr>
<tr>
<td>RESUME keyword</td>
<td>528</td>
</tr>
<tr>
<td>Accumulated database record bytes keyword</td>
<td>529</td>
</tr>
<tr>
<td>Return Codes</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Authorizing XIM procedures</td>
<td>697</td>
</tr>
<tr>
<td>Online Reorg function user exit routine</td>
<td>676</td>
</tr>
<tr>
<td>Reload function user exit routine</td>
<td>670</td>
</tr>
<tr>
<td>Unload function user exit routine</td>
<td>668</td>
</tr>
<tr>
<td>Index Build function return codes</td>
<td>619</td>
</tr>
<tr>
<td>Reload function return codes</td>
<td>619</td>
</tr>
<tr>
<td>Unload function return codes</td>
<td>618</td>
</tr>
<tr>
<td>Index record fields</td>
<td>611</td>
</tr>
<tr>
<td>Secondary Index Relationship Summary report fields</td>
<td>611</td>
</tr>
<tr>
<td>Segment Statistics report fields</td>
<td>613</td>
</tr>
<tr>
<td>Status of Segments Read from HD Unload Dataset report fields</td>
<td>613</td>
</tr>
<tr>
<td>Twin Segment Statistics report fields</td>
<td>614</td>
</tr>
<tr>
<td>User Exit Routine Action report fields</td>
<td>614</td>
</tr>
<tr>
<td>Variable Length Segment Statistics report fields</td>
<td>615</td>
</tr>
<tr>
<td>VL Segment Split Statistics report fields</td>
<td>616</td>
</tr>
<tr>
<td>Unload function return codes</td>
<td>618</td>
</tr>
<tr>
<td>Reload function return codes</td>
<td>619</td>
</tr>
<tr>
<td>Index Build function return codes</td>
<td>619</td>
</tr>
<tr>
<td>Share levels</td>
<td>640</td>
</tr>
<tr>
<td>Index Build function defaults by DBRC setting</td>
<td>643</td>
</tr>
<tr>
<td>Environment setup member options</td>
<td>649</td>
</tr>
<tr>
<td>Sample user exit routines</td>
<td>664</td>
</tr>
<tr>
<td>Register 1 requirements for Unload function user exit routine</td>
<td>667</td>
</tr>
<tr>
<td>Unload function user exit routine return codes</td>
<td>668</td>
</tr>
<tr>
<td>Register 1 requirements for Reload function user exit routine</td>
<td>669</td>
</tr>
<tr>
<td>Reload function user exit routine return codes</td>
<td>670</td>
</tr>
<tr>
<td>Online Reorg function user exit routine return codes</td>
<td>674</td>
</tr>
<tr>
<td>Register 1 requirements for Online Reorg function user exit routine</td>
<td>675</td>
</tr>
<tr>
<td>Online Reorg function user exit routine return codes</td>
<td>676</td>
</tr>
<tr>
<td>Authorizing XIM procedures</td>
<td>697</td>
</tr>
<tr>
<td>XIMCHECK return codes</td>
<td>704</td>
</tr>
</tbody>
</table>
About this book

This book provides information an IMS system programmer or database administrator should need to use the BMC MAXM® Reorg solutions, which include the following functions:

- Unload function (UNLOAD PLUS/EP)
- Reload function (RELOAD PLUS/EP)
- Index Build function (SECONDARY INDEX UTILITY/EP)
- Prefix Resolution/Update function
- Reorg function (FAST REORG FACILITY/EP)
- Image Copy function (during the Online Reorg or Reorg function)
- Online Reorg function (CHANGE RECORDING FACILITY)
- Online/Defrag function

This book assumes that you are familiar with IMS and JCL.

Like most BMC documentation, this book is available in printed and online formats. To request printed books or to view online books and notices (such as release notes and technical bulletins), see the support website at http://www.bmc.com/support.

**NOTE**

Online books are formatted as PDF or HTML files. To view, print, or copy PDF books, use the free Adobe Reader from Adobe Systems. If your product installation does not install the reader, you can obtain the reader at http://www.adobe.com.

The software also offers online Help. To access Help, press F1 within any product or click the Help button in graphical user interfaces (GUIs).

From the BMC Support Central website (http://www.bmc.com/support), you can

- download a zipped set of documentation PDFs from each product’s EPD page
- link to the BMC Documentation Center (https://webapps.bmc.com/infocenter/index.jsp) to browse documentation sets, or to view video demos (short overviews of selected product concepts, tasks, or features)
view individual product documents (books and notices) within the “A – Z Supported Product List”

You can order hardcopy documentation from your BMC sales representative or from the support site. You can also subscribe to proactive alerts to receive e-mail messages when notices are issued.

Conventions

This book uses the following special conventions:

- All syntax, operating system terms, and literal examples are presented in this typeface.

- Variable text in path names, system messages, or syntax is displayed in italic text:

  `testsys/instance/fileName`

- The symbol `=>` connects items in a menu sequence. For example, *Actions => Create Test* instructs you to choose the *Create Test* command from the *Actions* menu.

Summary of changes

This section summarizes changes to the functionality of MAXM Reorg Solutions for IMS and the IMS Database EP Utilities, listing the changes by product version and release date. The summary includes enhancements to the product and any major changes to the documentation.

Revision bars indicate where technical information has changed.

**Version 4.8 December 2011**

The following changes were made to the book:

- A third subparameter was added to the USEREXIT parameter. For more information, see “USEREXIT” on page 566.

- You can now use the SECINDEX parameter to specify HALDB secondary index partitions to build. For more information, see “SECINDEX” on page 534.

- All references to PRP partitioning, including the PRP partitioning section in Chapter 10, were removed.
Version 4.7  November 2010

The following changes were made to the book:

- Minor changes to sample JCL changing the region parameter to REGION=0M.
- Added “WFI BMP processing” on page 128.
- Added Chapter 9, “Copy function.”
- Added Chapter 12, “RECON Substitution Facility.”
- Added “HSAMOUT” on page 448

Version 4.7  May 2009

The following changes were made to the book:

- Updated the “Image Copy function” description on page 76 to indicate that dynamic allocation matrixes are supported when the Image Copy function is running under a reorganization function.
- Updated the descriptions of the FBFF and FSPF keywords in Table 5 on page 97.
- Updated the “Overview” section on page 109 of Chapter 3, “Online Reorg function” to indicate that you can use dynamic allocation models and matrixes when you run the Image Copy function under a reorganization function.
- Updated the “DD statements” section on page 116 of Chapter 3, “Online Reorg function.” for INPUT keyword updates.
- Corrected the description of PART keyword in the “Keywords” section on page 119.
- Updated the INPUT keyword in Table 9 on page 119.
- Updated the “AMSPDS DD” section on page 352 of Chapter 13, “JCL requirements.”
- Added the “FBFF” section on page 434 in Chapter 14, “User-controlled options.”
- Updated the “FSPF” section on page 441 in Chapter 14, “User-controlled options.”
- Updated the “IDCAMS” section on page 464 in Chapter 14, “User-controlled options.”
Summary of changes

- Updated the “INPUT” section on page 473 in Chapter 14, “User-controlled options.”

- Updated the Table 215 on page 511.

- Updated the “SNAPSHOT” section on page 543 in Chapter 14, “User-controlled options.”

Version 4.6.01  August 2008

The following changes were made to the book:

- Updated the Environment Setup Member Options in Appendix D, “Environment setup member.”

- Updated description of one combination of LIMITS and PART keywords that is not valid for HALDB databases in Table 215 on page 511.

Version 4.6.00  December 2007

The following changes were made to the book:

- Updated keyword exceptions for USERLOAD in the following section: “USERLOAD command” on page 237.

- Updated description to include LOAD in the following section: “FFTOHAL” on page 434.

- Updated the description of the SIZE subkeyword in the following section: “HIPO” on page 446.

- Updated the description of the MODEL keyword in the following section: “IDCAMS” on page 464.

- Added information on building selected secondary indexes from a WF1 file in the following section: “IMS/ESA partition support” on page 515.

- Updated the list of support limitations for HALDBs in the following section: “HALDB limitations” on page 688.
# Introduction

This chapter provides an overview of MAXM Reorg solutions. This chapter includes the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>36</td>
</tr>
<tr>
<td>MAXM Reorg/Online solution</td>
<td>37</td>
</tr>
<tr>
<td>MAXM Reorg/EP solutions</td>
<td>39</td>
</tr>
<tr>
<td>MAXM Reorg/EP Express solution</td>
<td>40</td>
</tr>
<tr>
<td>Comparison of utilities</td>
<td>40</td>
</tr>
<tr>
<td>Getting started</td>
<td>42</td>
</tr>
<tr>
<td>Goals of data management</td>
<td>42</td>
</tr>
<tr>
<td>Improve data availability</td>
<td>42</td>
</tr>
<tr>
<td>Conserve data resources</td>
<td>44</td>
</tr>
<tr>
<td>Preserve data integrity</td>
<td>46</td>
</tr>
<tr>
<td>Use familiar structure</td>
<td>47</td>
</tr>
<tr>
<td>MAXM Reorg/Online features and benefits</td>
<td>48</td>
</tr>
<tr>
<td>Reorg function features and benefits</td>
<td>49</td>
</tr>
<tr>
<td>Data set name swapping</td>
<td>51</td>
</tr>
<tr>
<td>Logically related databases</td>
<td>52</td>
</tr>
<tr>
<td>Unload function features and benefits</td>
<td>52</td>
</tr>
<tr>
<td>Reload function features and benefits</td>
<td>54</td>
</tr>
<tr>
<td>Index Build function features and benefits</td>
<td>55</td>
</tr>
<tr>
<td>Prefix Resolution and Update function features and benefits</td>
<td>58</td>
</tr>
<tr>
<td>Online/Defrag function features and benefits</td>
<td>60</td>
</tr>
<tr>
<td>DBA Toolkit features and benefits</td>
<td>61</td>
</tr>
<tr>
<td>Automated data collection</td>
<td>61</td>
</tr>
<tr>
<td>Automatic analysis</td>
<td>62</td>
</tr>
<tr>
<td>Automatic configuration</td>
<td>62</td>
</tr>
<tr>
<td>Conditional reorganization</td>
<td>62</td>
</tr>
<tr>
<td>Control block disassembly</td>
<td>62</td>
</tr>
<tr>
<td>DBD mapping</td>
<td>63</td>
</tr>
<tr>
<td>Exception management</td>
<td>63</td>
</tr>
<tr>
<td>JCL generator</td>
<td>63</td>
</tr>
<tr>
<td>Repository management</td>
<td>63</td>
</tr>
<tr>
<td>Database Management Console</td>
<td>64</td>
</tr>
<tr>
<td>Statistical information</td>
<td>64</td>
</tr>
</tbody>
</table>
Overview

This section provides a general overview of database reorganization and describes the MAXM Reorg solutions.

Reorganization is a periodic requirement of database management. If you have perfectly tuned databases and you never update them, you will never need to reorganize them. But you probably update your databases regularly and, therefore, you need to reorganize them from time to time.

As information is added, updated, and deleted, a database becomes physically disorganized, decreasing operating efficiency. More I/O operations are needed to retrieve a segment and its dependents when they are physically disorganized than when they are physically adjacent. When this occurs, response time slows noticeably, with a corresponding decrease in end-user productivity.

A database must be reorganized periodically to improve operating efficiency and to implement changes. Physical reorganization involves placing each root segment together with its dependent segments into one block (or into adjacent blocks if they do not fit into a single block) so that fewer I/O operations are needed to retrieve the root and its dependents. Any secondary indexes for the reorganized database must be rebuilt, and logical relationship pointers between databases must be resolved and updated. To enable recovery of the reorganized database, an image copy is required after the reorganization.

As the size and complexity of a database increases, reorganization processing time increases. Traditional reorganization utilities force a database to be offline during reorganization, and a sufficiently large window of down time must be found for the reorganization. You might not be able to find this window of down time, or you might not be able to take advantage of secondary indexes and logical relationships because of the additional reorganization requirements.

The need for an online reorganization utility has become more urgent as more and more installations strive to be available 24 hours a day, seven days a week.
MAXM Reorg/Online solution

MAXM Reorg/Online is the total solution for database reorganization. The solution includes all of the functionality that you need to perform online and near-online reorganizations for performance improvement.

MAXM Reorg/Online includes the following functions:

— Online Reorg function
— Reorg function
— Unload function
— Reload function
— Index Build function
— Copy function
— Image Copy function
— Prefix Resolution and Update function
— Online/Defrag function
— DBA Toolkit
— Non-disruptive maintenance

The Online Reorg function provides a near online reorganization for full-function databases. The Online Reorg function lets you reorganize an online database and retain full access. You can update the database while it is being reorganized. The Reorg function manages the reorganization. While the database is being reorganized, the Online Reorg function captures the updates and applies the changes to the reorganized database when the reorganization is complete. The Image Copy function is invoked to take an image copy of the database as it is being reorganized. The database must be offline for a short time at the beginning and the end of the reorganization.

The Reorg function provides the ability to invoke the Image Copy function to take an image copy of the database as it is being reorganized. The Reorg function also lets you reorganize a database and have read access to the database while it is being reorganized. The Reorg function invokes the other functions to reorganize the database. The database must be off-line for a short time (usually less than one minute) to swap data set names. The Prefix Resolution and Update function in the Reorg function manages logical relationships.

The Unload function replaces the IMS HD Reorganization Unload utility, the IMS Database Scan utility, and HSSR.

The Reload function replaces the IMS HD Reorganization Reload utility.
The Index Build function replaces the six IMS utilities that are required to build secondary indexes. The Index Build function provides the ability to build primary and secondary indexes in the same job step, with one command and one pass of the database.

The Image Copy function takes an image copy of the database as it is being reorganized.

The Prefix Resolution and Update function is invoked by the Online Reorg function (via the Reorg function). The Prefix Resolution and Update function also invokes the Image Copy function. The Prefix Resolution and Update function replaces the IMS Prefix Resolution and Prefix Update utilities and includes several optional functions.

The Online/Defrag feature is innovative technology that offers customized analyses so that you can identify and reorganize only those areas of a database that need to be reorganized. The Online/Defrag feature is also the only function that keeps your database online—all day, every day. Online/Defrag does not replace traditional reorganizations. You must still perform regular reorganizations. Online/Defrag is a supplemental function that provides targeted online reorganizations.

The DBA Toolkit offers JCL generation and historical analysis features. The JCL generator lets you automatically generate and submit optimized JCL to perform reorganization tasks for full-function IMS databases. The historical analysis feature displays historical data about your IMS database data sets and segments so that you can identify and predict database trends.

Figure 1 shows how the standard functions work together.
BMC took the first step toward an online reorganization with the introduction of the Reorg function. The Reorg function creates a "shadow" of a database during reorganization and allows read capabilities during almost the entire reorganization. You can make some database description (DBD) changes during the reorganization. A short outage is needed to swap the data set names after the reorganization and to make the reorganized database available. The Reorg function includes a Parallel Image Copy feature. The only missing function was the ability to update the database while its shadow is being reorganized. The Online Reorg function provides this missing piece.

The Online Reorg function is an extension of the Reorg function that lets you update a database while it is being reorganized. The Online Reorg function does not write before and after log records for the shadow database; this keeps traffic away from the log data sets. The Online Reorg function requires a short outage to swap the data set names. Complicated scenarios, such as manually stopping and restarting batch message processing programs (BMPs), doing online changes after a DBD change, or other unusual processing, might cause the outage to be longer.

DBRC control is essential to the integrity and success of the reorganization process. The Online Reorg function and Online/Defrag function require DBRC registration. For more information, see Appendix C, “DBRC support.”

**MAXM Reorg/EP solutions**

MAXM Reorg/EP solutions include all of the functions that are offered in MAXM Reorg/Online except the Online Reorg function. MAXM Reorg/EP solutions includes the following functions:

- Reorg function
- Unload function
- Reload function
- Index Build function
- Copy function
- Image Copy function
- Prefix Resolution and Update function
- Online/Defrag function (optional feature)
- DBA Toolkit
MAXM Reorg/EP Express solution

The newest addition to the MAXM Reorg solutions is MAXM Reorg/EP Express for IMS (MXP). The solution is designed to offer a simple and cost effective way to upgrade to the Extended Performance (EP) utilities. MXP includes the following functions:

— Reorg function
— Unload function
— Reload function
— Index Build function
— DBA Toolkit

Comparison of utilities

MAXM Reorg functions offer the following features that are not available in the classic database utilities or IMS.

■ The Online Reorg function offers:

— multiple database reorganizations
— near-online reorganization
— prefix resolution and update support for logically related databases
— limited HALDB support

■ The Reorg function offers:

— multiple database reorganizations
— improved I/O techniques
— dynamic allocation of database data sets
— parallel processing
— prefix resolution and update support for logically related databases
— limited HALDB support

NOTE

The Image Copy function is not available in the Reorg function with MAXM Reorg/EP Express.
The Unload function offers:

— improved I/O techniques
— dynamic allocation of database data sets
— database restructuring
— HALDB support

The Reload function offers:

— improved I/O techniques
— dynamic allocation of database data sets
— parallel processing
— segment positioning
— HD Sort utility

The Index Build function offers:

— improved I/O techniques
— parallel processing
— ability to build primary and secondary indexes simultaneously
— dynamic allocation of database data sets
— automatic selection of best scan method for database
— simplified command structure
— HALDB support

The Online/Defrag function offers:

— reorganizations with no downtime
— customized record analyses
— targeted reorganizations
— reorganization parameter recommendations
— limited HALDB support

Each function enables you to complete specific reorganization tasks. For example, if you only need to build indexes for a database, you can run the Index Build function as a stand-alone utility.

The Reorg function enables you to reorganize one or more databases and retain read access during the reorganization. The Reorg function manages the reorganization by using the Unload function, the Reload function, the Index Build function, and the Image Copy function. The Reorg function also integrates prefix resolution and update for logical relationships.
The Online Reorg function enables you to reorganize one or more databases and retain complete access during the reorganization. When you specify the SHARE(YES,UPDATE) keyword with the REORG command in an Online Reorg function job, you can reorganize the database while it is being updated. An image copy is taken automatically as the reorganization progresses. MAXM Reorg/Online offers the most automated online reorganization possible.

The Online/Defrag function completely eliminates downtime. You can reorganize specific records or your entire database and stay online for concurrent updates.

**Getting started**

BMC used new technology to provide MAXM Reorg/Online. The most dramatic technological change is in the Online Reorg function.

Installing the Online Reorg function involves a distinctly different set of considerations and possibly the involvement of other individuals, such as IMS system programmers. For more information about installing the Online Reorg function, see the *BMC Products for IMS Installation Guide*.

**Goals of data management**

This section describes critical goals that data managers must achieve:

- improve data availability
- conserve data resources
- preserve data integrity
- use familiar structure

MAXM Reorg solutions help you meet these goals while making only minimal changes to the way that you perform reorganizations.

**Improve data availability**

Data availability is critical. Companies must provide a reasonable response time for interactive processing. Many companies must also support extended hours of operation, sometimes around the clock, seven days a week. The large volume of data that these companies must manage further complicates the challenge of maintaining data availability, because batch processes run longer and longer as they handle more and more data.
Given the need for accessible data, one goal of data management is to improve data availability. Data managers must strive to maintain or improve the speed of data processing and reduce planned and unplanned down time.

MAXM Reorg solutions greatly improve data availability by significantly reducing the time required to reorganize a full-function IMS database. When you use the MAXM Reorg functions separately, each function decreases the total reorganization time and allows you to get the database online faster.

The Online/Defrag function in MAXM Reorg solutions completely eliminates downtime. With the Online Reorg function in MAXM Reorg/Online, the database is unavailable for a very brief time.

With a traditional reorganization, you must take the database offline for the duration of the reorganization. MAXM Reorg/Online enables you to have complete access during the unload, reload, index build, and image copy processes. Access to the database is restricted only during startup and data set name swapping.

MAXM Reorg solutions increase data availability by offering the following features and benefits:

- **Speed**

  MAXM Reorg functions run significantly faster than the IMS utilities and BMC utilities that they replace. The EP functions run up to twice as fast as their counterparts:

  - To speed the reorganization process, the Unload function uses advanced I/O techniques or native OS/VS access methods rather than IMS DL/I to read the database, reducing the number of I/Os issued to retrieve database segments.

  - The Reload function uses special I/O routines to optimize the I/O throughput during the reload step, decreasing elapsed time.

  - The Index Build function enables you to build primary and secondary indexes in the same job step, saving multiple job steps. It also provides a Sequential Scan feature that enables the database to be scanned much faster than with a traditional hierarchical scan.

  - The Reorg function passes records rather than moving them to an unload file. Real storage usage does not increase because segments are passed directly, not stored in hyperspace. This speeds the reorganization process considerably, saving you time and money.

- **Reorg function parallel processing**

  The MAXM Reorg functions can run in parallel mode, using subtasks within a single MVS image to decrease elapsed time.
Conserve data resources

- improved database performance

A disorganized database typically results in poor overall performance. Although disorganized databases degrade performance, some companies are restricted from reorganizing databases because of the time involved. MAXM Reorg solutions significantly reduce reorganization time (and database down time), so you can reorganize as often as needed to improve database performance.

- no downtime

The Online/Defrag function completely eliminates downtime. You can reorganize trouble spots or your entire database and stay online for concurrent updates.

Conserve data resources

Few companies have unlimited resources for managing their data. It is usually difficult to find enough time, money, hardware, and people to handle the normal work load and the special projects that are assigned periodically. If an emergency occurs, carefully planned schedules and budgets can be destroyed without the correct preparation.

A goal of data management is to conserve data management resources, using them wisely and without waste. Data managers must use techniques that enable them to accomplish the goals of data integrity and data availability within the limitations of budgets, deadlines, and personnel.

MAXM Reorg solutions help conserve data resources by offering the following features and benefits:

- simplified reorganizations

An entire physical database, including all primary and secondary indexes, can be reorganized in one step. Whether you choose the Online Reorg function or the Reorg function, you submit one set of JCL that runs the entire reorganization process. You no longer need to supply JCL for individual functions.

Because the original database remains intact during reorganization, you can eliminate the precautionary image copy before the reorganization.
• simplified JCL

All utilities are included in one load library. The functions can dynamically allocate many of the data sets needed for processing. You can use your existing BMC JCL; just change some keywords in the PLUSIN control statement data set. The Online Reorg function does not require a separate command; to invoke the Online Reorg function, include the SHARE(YES,UPDATE) keyword in the PLUSIN control statement data set. For more information, see Chapter 13, “JCL requirements.”

• less decision making

MAXM Reorg solutions can choose the optimum methods for accomplishing many tasks:

— The Unload function uses self-tuning buffers. The Unload function continues to acquire buffers as long as it detects a need for more buffers. When the maximum number of usable buffers is reached, the Unload function stops acquiring more buffers.

— The Reload function uses buffers as needed—you don’t need to specify any buffers.

— Some keywords include an AUTO value, which allows the utility to choose the best method to accomplish the task. For example, you can use SCAN(AUTO) to instruct the Index Build function to choose the most efficient scanning method.

• expanded control

You can run the functions with the defaults, or you can set options to customize the reorganization process. For more information, see Chapter 14, “User-controlled options.”

• improved I/O techniques

Each function uses improved I/O techniques that save CPU time.

• reduced CPU usage

MAXM Reorg solutions use significantly fewer CPU cycles than the IMS utilities they replace. This reduced CPU usage frees resources for other jobs in the system. You can set options to further reduce CPU time. For example, you can instruct the Reorg function to not expand and recompress segments; however, the Reload function must expand a segment if it needs to create index maintenance records.
perform custom record analyses to target reorganizations

The Online/Defrag function performs custom record analyses so that you can identify and reorganize only those records that are causing database performance problems. Targeting your database reorganization conserves data, system, and human resources.

Preserve data integrity

A company’s data is critical to its day-to-day operation and its ongoing success. Lost, incorrect, or incomplete data could have disastrous consequences for the company and its customers.

Perhaps the most important goal of data management is to preserve data integrity. Although IMS is a highly reliable data management system, any system is vulnerable to procedural, hardware, and software errors. Data managers must keep these errors from destroying vital data.

MAXM Reorg solutions help preserve data integrity by providing the following features and benefits:

■ enhanced data integrity

The Reorg function checks the JCL for logic errors. The database data sets that are used by the unload process must contain data, and those used by the reload must be empty. The Reorg function checks for these conditions, increasing reliability and preventing errors such as overwriting data.

■ recoverability

When you use the Online Reorg function or the Reorg function to reorganize a database, your original database is preserved and you can revert to it in the unlikely event of a failure. In a traditional reorganization, you probably delete the database data sets after the unload process is complete. If a failure occurs during load processing, you are forced to restart the load process or begin an image copy restore (or full recovery) before you can make the database available again. You don’t need to recover the database (because the original data sets still exist).
Use familiar structure

BMC designed MAXM Reorg solutions to be compatible with familiar reorganization products. Converting to these solutions is a simple process because the following features and benefits are provided:

- **compatibility with corresponding IMS utilities, BMC classic database utilities, and MAXM Reorg for IMS.**

  MAXM Reorg solutions require few (or no) changes to your existing JCL. Simply install and run the functions, which saves time in preparing and debugging JCL. To use the new features, just add keywords to the PLUSIN control statement data set. You can include optional statements to specify work data sets.

- **familiar reports**

  Each function generates standard reports. The reports provide database statistics and can be a valuable aid in analyzing the database, allowing you to make intelligent changes to the DBD or to set options that enhance performance. You can view the reports online and/or obtain a hardcopy.

  The Online Reorg function and the Reorg function generate additional reports that summarize the entire reorganization. For more information, see Chapter 15, “Reports.”

- **IMS/ESA Partition Support**

  MAXM Reorg solutions support partitioned and non-partitioned databases. You can use the IMS/ESA Partition Support product to reorganize selected partitions by using the Reorg function, Unload function, Reload function, and Index Build function. With the Online Reorg function, you must reorganize all database partitions at the same time. With the Prefix Resolution and Update function, you must update and resolve prefixes for all database partitions at the same time.

- **BMC support for HALDBs**

  BMC offers support for HALDBs. Support includes a simple process to migrate a full-function database to a HALDB. In addition, you can use the Unload function, Reload function, Index Build function, Reorg function, and Online Reorg function to reorganize HALDBs.

- **BMC Customer Support**

  One of the best features of MAXM Reorg solutions is the company that stands behind them. A BMC Customer Support representative is available 24 hours a day, seven days a week, to answer questions and solve problems. Any time you need support, call the toll-free number listed on the inside front cover of this manual or call your local support center.
MAXM Reorg/Online features and benefits

MAXM Reorg/Online enables you to reorganize a database while users have read and update access to the database.

MAXM Reorg/Online offers the following features and benefits:

- **complete access during reorganization**
  
  You can allow online applications to have complete access (read and update) to the database during most of the reorganization process. The Online Reorg function records changes and applies those changes to the reorganized database. The database is offline only to swap the data set names.

- **automated database reorganization**
  
  MAXM Reorg/Online automates the database reorganization process by eliminating virtually all manual intervention. The Online Reorg function, by calling the other functions, can dynamically allocate most data sets required for the reorganization, including the database data sets. The Online Reorg function invokes the Image Copy function to take an image copy of the reorganized database automatically. The Online Reorg function invokes the prefix resolution and update function. You can request automatic data set name swapping. All DBRC notifications are done for you. If BMC’s APPLICATION RESTART CONTROL (AR/CTL™) is installed, any BMPs under AR/CTL control are suspended and restarted automatically. This automation reduces the risk of errors.

  If the BMC POINTER CHECKER PLUS® product is installed, the Image Copy function can invoke the POINTER CHECKER PLUS Hash Checking utility to validate pointers while an image copy is being created.

- **non-disruptive maintenance**
  
  You can update the Change Recording Facility (CRF) and Online Defrag (OLR) components that execute in the IMS control region without requiring that the IMS control region be recycled.
Reorg function features and benefits

The Reorg function has these primary goals:

- simplify the reorganization process
- speed the reorganization of very large databases
- increase database availability while allowing read access to the database

The Reorg function initiates and coordinates the execution of the Unload function, Reload function, Index Build function and the Prefix Resolution and Update function, increasing the speed of reorganization by overlapping execution of those utilities and eliminating a large percentage of EXCPs. The Reorg function eliminates the need for an unload file by passing records from the Unload function directly to the Reload function without writing physical records. The Reload function loads the records that have been passed to it from the Reorg function directly to a new database data set or, if requested, creates an image copy.

If the database has secondary index records, the Reorg function passes the secondary index information to the Index Build function for processing. The Index Build function rebuilds secondary indexes for the reorganized database. The DFSURWF1 file may be eliminated if there are no logical relationships, or if the prefix resolution and update function is integrated into the Reorg function. The elimination of EXCPs saves CPU time and total run time. The savings in total execution time can be significant.

You can use the ICP(YES) keyword to invoke the Image Copy function. An image copy is taken of the database as it is reorganized.

When the reorganization is complete, the data set names of the old and new files must be exchanged (swapped) so the reorganized database data sets can be made available. The Reorg function invokes IDCAMS to accomplish the swapping process and notifies DBRC that a reorganization has taken place. You can perform these tasks manually, or you can execute the RESTART command in a separate step following the reorganization.

If you are reorganizing a database that has logical relationships, you are responsible for providing the DFSURCDS information as created by the IMS Prereorganization utility. The prefix resolution and update function might be required. In most cases, the Reorg function can integrate the prefix and resolution update function to perform these tasks automatically. If the process is not integrated, you are responsible for executing the prefix resolution and update function before the swap process.

If the reorganization terminates before the data set names are swapped, you can abandon the reorganization. This eliminates the need to take an image copy of the disorganized database before the reorganization and the need to recover if the reorganization fails.
With the Reorg function, you can keep the database online and allow *read* access to users during the reorganization. The database must be offline only long enough to swap the data set names. The reorganization process can be automated so that no manual intervention is required.

The Reorg function speeds and simplifies a database reorganization. The Reorg function offers the following additional features and benefits:

- **parallel image copy**

  The Reorg function can invoke the Image Copy function automatically to take an image copy of the reorganized database. For more information, see “ICP” on page 458.

  If the BMC POINTER CHECKER PLUS® product is installed, the Image Copy function can invoke the POINTER CHECKER PLUS Hash Checking utility to validate pointers while an image copy is being created.

- **data set name swapping**

  At the end of the reorganization process, the Reorg function can swap data set names automatically. The Reorg function prohibits authorization (locks the database) during the swap process. If you specify automatic data set name swapping, DBRC is required. For more information, see “Data set name swapping” on page 51.

- **read access during reorganization**

  With a traditional reorganization, you must take the database offline for the duration of the reorganization. The Reorg function enables you to allow *read* access during the unload, reload, and index build processes. Access to the database is restricted only during data set name swapping. The SHARE keyword controls access to the database during reorganization. For more information, see “SHARE” on page 540.

- **DBD change during reorganization**

  The Reorg function allows you to specify different DBD libraries for use by the Unload function and the Reload function. This allows you to change many of the characteristics of the DBD across a reorganization. For more information, see “OLDDBD” on page 496.

- **reduced EXCPs**

  The unload data set (DFSURGU1) is bypassed. This reduces EXCPs, and no tape drive or DASD work space is needed to store the data set.
Index maintenance records are not written to DFSURWF1. This reduces (or eliminates) the space and the EXCPs for that file.

If the prefix and resolution update function is integrated into the reorganization, logical relationship records are not written to DFSURWF1/PRPURWF1 or DFSURWF3. This reduces (or eliminates) the space and the EXCPs for these files.

**Data set name swapping**

Automatic data set name swapping is an important feature. When the reorganization is complete, you can allow the Reorg function to give the reorganized database data sets the names of the active database data sets (as registered to DBRC).

Because the Reorg function copies the database to new files, the database can still be accessed by an online system or batch jobs (with read access). Upon completion of a successful reorganization, the Reorg function puts the database in exclusive mode to swap the data set names.

During data set name swapping, the database must be available exclusively to the reorganization process. The Reorg function issues the NOTIFY.REORG command to inform DBRC of the reorganized database status.

Automatic data set name swapping is available only when all of the following conditions are met:

- DBRC must be active, and the database (and its index databases) must be registered. If you do not plan to recover the index databases, they must be registered to DBRC and set to NONRECOV.

- The names of the database data sets being unloaded must be registered to DBRC.

- The data set names of the output (shadow) files must not be registered to DBRC. There is one exception: If you are not using SHARE(YES), the index data sets that are registered to DBRC can be used as the output index data sets. They are excluded from data set name swapping because they already have the correct, registered names.

- The Reorg function must be able to obtain exclusive access to the database.

If you choose automatic swapping and other processes are still using the database in read mode, the Reorg function issues a write-to-operator with reply (WTOR) request to stop the database with a /DBR. When the swap is completed, the Reorg function issues a write-to-operator (WTO) message stating that the database is available. You might want to include these messages in a message processing facility (such as the BMC AutoOPERATOR™ product) to further automate the process.
You can save DASD space by specifying SWAP(x,IC). For more information, see “SWAP” on page 548.

Logically related databases

The Reorg function integrates the prefix resolution and update function if WF1 is not specified. For more information about the Reorg function and logical relationships, see “Reorganization for logically related databases” on page 86.

Unload function features and benefits

The Unload function replaces IMS HD Reorganization Unload utility (DFSURGU0). The Unload function functionally replaces the HISAM Reorganization Unload utility (DFSURUL0). To use the Unload function instead of DFSURUL0, you must change the DFSURUL0 JCL to the HD Unload (DFSURGU0) format. For more information, see Chapter 13, “JCL requirements.”

The Unload function reads a database and writes records to an HD Unload data set in hierarchical order. You can use this data set to reload the records in hierarchical order. When records are in order, database processing improves because fewer I/O operations are needed to retrieve data.

The Unload function offers the following features and benefits:

- **HALDBs**
  
  You can use the Unload function to unload a HALDB.

- **segment selection**
  
  You can unload all segments or only selected segments by using the SEGSEL keyword. You can specify as many as 30 segment types. For more information, see “SEGPOSO” on page 536.

- **decreased DL/I application program run time**
  
  An application program interface (API) allows batch DL/I application programs that use GET-type calls to use the facilities of the Unload function to retrieve segments faster than DL/I retrieves them. Programs that read large portions of a database sequentially might realize significant reductions in elapsed time and CPU usage. You do not have to recompile or relink-edit the application program to use the API. For more information, see “API” on page 199.
unload function features and benefits

- ability to retrieve selected records

The Database Selection facility (DBSF) allows you to retrieve selected records from a database. You can print the records or write them to an HD Unload data set. The primary use of the DBSF is to enable you to create test databases. For more information, see “Database Selection facility” on page 192.

- user exit facility for processing each segment

The Unload function and the Reload function provide a user exit facility. For more information, see “USEREXIT” on page 566.

- HDAM-to-HIDAM migration and PHDAM-to-PHIDAM migration

The USEINDEX keyword, used with the Index Build function, lets you optimally convert an HDAM or PHDAM database to an HIDAM or PHIDAM database, respectively. This eliminates the following traditional requirements:

- application program to unload the HDAM or PHDAM database and build an appropriate sort key

- sorting of segments into key value sequence

- application program to reload the HIDAM or PHIDAM database

- ability to continue processing after a sequence error

The Unload function always sequence-checks twin chains. If the Unload function detects a sequence error, it terminates by default. The SEQERROR keyword allows you to specify that the Unload function continue with the unload. When the unload has ended, the Unload function gives you the appropriate data to correct the sequence error.

- ability to fix broken/invalid physical twin forward (PTF) and physical twin backward (PTB) root pointer chains

The USEINDEX keyword enables you to correct a broken/invalid PTF or PTB root pointer chain by forcing the roots to be accessed from an index:

- For HIDAM and PHIDAM databases, you can bypass bad PTF and PTB pointers on the root segment by forcing the Unload function to unload by using the primary index. Reloading the database corrects the pointer errors. If the primary index is invalid, you might be able to rebuild it by using the Index Build function.
For HDAM and PHDAM databases with a secondary index whose target is the root segment and uses the root segment key, you can use the Unload function to unload by using the secondary index. You can reload the database to reconstruct the database pointers. If the database does not have a secondary index, you can define one temporarily and use the Sequential Scan feature of the Index Build function.

For more information, see “USEINDEX” on page 564.

Reload function features and benefits

The Reload function replaces IMS HD Reorganization Reload utility (DFSURGL0).

The Reload function reloads records that were unloaded earlier in a reorganization, using the HD Unload file as input. The Reload function writes any secondary index and logical relationship records to a DFSURWF1 or PRPURWF1 file that can be used for subsequent processing.

The Reload function offers the following features and benefits:

- HALDBs

  You can use the Reload function to reload a HALDB. Extended Pointer Set (EPS) pointers are automatically healed during the reload process.

- support for logical relationships and secondary indexes

- ability to initialize an empty database without a LOAD program specification block (PSB)

- ability to reload compressed databases without a special DBD

- user exit facility for processing each segment

  The Reload function and the Unload function provide a user exit facility. For more information, see “USEREXIT” on page 566.

- ability to load a database from an HSAM database
ability to continue processing after a sequence error

The Reload function allows you to continue after a sequence error. You can specify a limit on the number of sequence errors that occur before the Reload function abends. If you continue with the reload, the Reload function provides the appropriate data to correct the sequence error. For more information, see “SEQERROR” on page 537.

an interface to user load programs and initial load of a database faster than IMS

The Reload function includes an application program interface (API) that provides a transparent linkage between a user-written DL/I load program and the Reload function. For more information, see “API” on page 234.

ability to sort data into physical sequence

All unload utilities unload data into hierarchical sequence. The HD Sort utility allows you to sort unloaded data into physical sequence before you reload the data. When the data is in physical sequence, it reloads faster. The Reload function required the data to be in physical sequence. You can run the HD Sort utility in a separate step before you reload the data or as part of the Reload function job. For more information, see “HD Sort utility” on page 227 and “HDSORT” on page 443.

---

**Index Build function features and benefits**

The Index Build function reduces the IMS multi-step index build process to a one-step process. Because the Index Build function creates a primary or secondary index much faster than the IMS Reorganization Unload/Reload utilities, the primary (target) database is offline for less time.

Because the Index Build function can sort index work records and create indexes in parallel mode, elapsed time is reduced even more. The Index Build function is easy to use—it requires only one command.

The Index Build function replaces the following utilities and programs that are required to create and maintain index databases:

- Prereorganization (DFSURPR0)
- HD Reorganization Unload (DFSURGU0)
- HD Reorganization Reload (DFSURGL0)
- Prefix Resolution (DFSURG10)
- HISAM Reorganization Unload (DFSURUL0)
- HISAM Reorganization Reload (DFSURRL0)
The Index Build function provides most of the functionality available in SECONDARY INDEX UTILITY, and it provides added functions.

The Index Build function performs the following functions:

- creates a new secondary index for an existing database
- maintains or rebuilds an existing secondary index
- rebuilds a primary index for a HIDAM or PHIDAM database

The Index Build function generates several comprehensive reports that provide information about index availability and performance.

The Index Build function offers the following features and benefits:

- support for HALDBs

  Index Build function supports PHIDAM databases. You can use the stand-alone Index Build function to build primary and secondary indexes for a HALDB. You can specify selected partitions.

- Index Build function does not require the healing process thereby reducing overhead.

  If a database has been opened for update, the healing process heals the pointers and any subsequent reference will not have to heal the pointer. If the database is opened without update (read-only), the healing occurs but the database is not updated; therefore, all subsequent references must heal the pointer again.

- ability to build the primary index for HIDAM and PHIDAM databases and the secondary indexes in the same job step

  The Index Build function enables you to simplify your JCL by building multiple secondary indexes in one job step rather than in several separate steps. Only one command is required.

- ability to build the primary index for a HIDAM or PHIDAM database and the secondary indexes in the same job step

  The Index Build function can rebuild the primary index for a HIDAM or PHIDAM database that has been destroyed. It takes less time for the Index Build function to rebuild the primary index than it does to reorganize the entire database or to recover the primary index from an image copy and apply the logs.

  You can build all indexes for a database in the same job step, with one pass of the data.
ability to scan a database sequentially rather than hierarchically

The Index Build function determines the most efficient scanning method for a database. If possible, the Index Build function reads blocks sequentially, identifies segments, and extracts data needed to build an index. This sequential scan can run significantly faster than a traditional hierarchical scan.

ability to create empty secondary indexes for use with empty databases

If you need to create a secondary index for an empty database, you can simplify the process by using the Index Build function.

simplified command and keyword structure

The Index Build function has only one command: BUILD. You can scan the database and build the index in the same job step with the same BUILD command; you no longer need to specify XSCN and XLOD commands. The Index Build function recognizes most SECONDARY INDEX UTILITY commands and keywords; it converts the commands and keywords to the appropriate Index Build function equivalents and writes a report that explains which commands and keywords were converted.

elimination of image copies of index databases

Most installations perform frequent, scheduled image copies of index databases, even if they do not need to recover those databases. Because the Index Build function can rebuild lost or damaged indexes by using only the primary database as input, image copies of indexes are no longer needed for recovery. You can avoid the overhead and expense of these image copies without compromising data recoverability.

ability to sort and load indexes in multiple address spaces

You can save elapsed time by sorting and loading indexes in separate address spaces. When you specify SIUSORT(E), the Index Build function creates a separate address space for each index. While the Index Build function job is running, you will see a separate MVS job for each index. When all jobs are complete, the output is merged into the original job. For more information, see “SIUSORT” on page 542.
Prefix Resolution and Update function features and benefits

The Prefix Resolution and Update function replaces the IMS Prefix Resolution and
Prefix Update utilities and provides significant performance improvements, even
when you have finely tuned the IMS utilities.

The prefix resolution and update function processes logical relationships up to ten
times faster than IMS utilities. The prefix resolution and update function offers the
following features and benefits:

- You can install and implement the Prefix Resolution and Update function for
  immediate performance improvements. The Prefix Resolution and Update
  function is compatible with IMS JCL.

- Reduces system resource and sort work space requirements.

- Reduces cost and processing time by restarting jobs between phases if an abend
  occurs.

- Improves error handling when error conditions occur. Recorded information
  assists with problem diagnosis. You can avoid unnecessary abends when the error
  condition is acceptable and avoid further processing when the error is not
  acceptable.

The Prefix Resolution and Update function resolves logical relationships and updates
prefix information during the reorganization of logically related IMS databases. The
Prefix Resolution and Update function provides job and statistical reports, and it
offers an ISPF interface that you can use to set product options and to view and
maintain reports.

The following functions are available:

- The IMS-compatible Prefix Resolution function runs two to four times faster and
  uses 40 percent less sort work space than the IMS Prefix Resolution utility. For
  more information, see “IMS-compatible Prefix Resolution function” on page 286.

- The High-performance Prefix Resolution function runs up to 10 times faster than
  and uses about the same amount of sort work space as the IMS Prefix Resolution
  utility. For more information, see “High-performance Prefix Resolution function”
  on page 272.

- The Two-step Prefix Resolution function separates the Parent-Child resolution
  phase from the Logical Twin resolution phase, allowing restart of a failed job
  between phases. For more information, see “Two-step Prefix Resolution function”
  on page 289.
The Prefix Update function can be run as a stand-alone job or concurrently during any prefix resolution function. Stand-alone Prefix Update provides significant performance improvements over the IMS Prefix Update utility. The Concurrent Prefix Update function provides a 10 percent performance improvement over the Stand-alone Prefix Update function. For more information, see “Stand-alone Prefix Update function” on page 295 and “Concurrent Prefix Update function” on page 277.

The Secondary Index Resolution function provides compatibility with the IMS Prefix Resolution function by sorting the secondary index work file records during the prefix resolution step or in a separate job step. For more information, see “Secondary Index Resolution function” on page 297.

The Work File 1 (WF1) Generator has interfaces to the IMS and BMC utilities that create logical relationship work file records. It can write BMC–format records, which are processed more efficiently than IMS-format records. You can create BMC–format records during load, reload, and scan processes.

**NOTE**

BMC–format records are not compatible with IMS-format records. BMC–format records can be processed by the Prefix Resolution and Update function only.

The WF1 Generator lets you separate secondary index work file records from logical relationship work file records. For more information, see “Work File 1 Generator” on page 300.

Work File Splitter. The Work File Splitter processes the DFSURWF1 work file, separating secondary index work file records from logical relationship work file records. For more information, see “Work File Splitter” on page 304.

The Prefix Resolution and Update function offers the following features and benefits:

- **orphan handling**
  
  You can define whether logical parent or logical child orphans in the work file will cause an abend. You can print information about any orphans that are found.

- **fixed-length sorting**
  
  You can define the work file records as fixed-length. Performance will improve, but more sort work space might be required.
work file record handling

You can create and use BMC-format work file records. The BMC-format records are shorter, so less sort work space and fewer I/O operations are required to process them. The Prefix Resolution and Update function describes the work file records more accurately to the sort, so logical relationship and secondary index records are sorted more efficiently.

Online/Defrag function features and benefits

Online/Defrag provides the following features and benefits:

- freedom to choose to reorganize your entire database or only those parts of the database that need to be reorganized:
  - reorganizes specific records, based on customized record analyses.
  - scans the entire database and reorganizes every record that meets specified criteria.
  - reorganizes your entire database, block by block.

- keeps your database online during reorganization:
  - remains online all day, every day—no down time
  - eliminates need for batch windows

- sophisticated record analyses to identify records that are good candidates for a targeted reorganization

- recommendation for the optimal reorganization parameters

- keywords that enable you to specify reorganization parameters and customize record analyses

- optimization of database reorganization efficiency:
  - You can target your reorganization at fragmented records that are affecting database performance.
  - Your database reorganization costs decrease.
— You can reorganize more often. Reorganization improves database performance, which improves online response time and end-user productivity.

— Minimal resources are required.

- no need for additional direct access storage device (DASD) to hold shadow copies of your database during conventional concurrent reorganizations
- no need to perform an image copy after the reorganization, because all updates are logged to IMS standards

### DBA Toolkit features and benefits

Table 1 lists the DBA Toolkit features that are available with the MAXM Reorg for IMS solutions.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>automated data collection</td>
<td>page 61</td>
</tr>
<tr>
<td>automatic configuration</td>
<td>page 62</td>
</tr>
<tr>
<td>control block disassembly</td>
<td>page 62</td>
</tr>
<tr>
<td>JCL generator</td>
<td>page 63</td>
</tr>
<tr>
<td>repository management</td>
<td>page 63</td>
</tr>
<tr>
<td>Database Management Console</td>
<td>page 64</td>
</tr>
<tr>
<td>statistical information</td>
<td>page 64</td>
</tr>
</tbody>
</table>

### Automated data collection

MAXM Reorg/Online and MAXM Reorg/EP with Online/Defrag Feature Utilities lets you define schedules to gather and analyze data.

**NOTE**

The repository management feature is available with all MAXM Reorg for IMS solutions.
Automatic analysis

MAXM Reorg/Online and MAXM Reorg/EP with Online/Defrag Feature Utilities automatically analyzes your databases to identify current problems and predict future problems.

Automatic configuration

The automatic configuration makes it easy to get started with MAXM Reorg/Online and MAXM Reorg/EP with Online/Defrag Feature Utilities by building the required components in the Databases window and populating suggested default parameter values.

NOTE
The automatic configuration feature is available with all MAXM Reorg for IMS solutions.

Conditional reorganization

MAXM Reorg/Online and MAXM Reorg/EP with Online/Defrag Feature Utilities saves time and resources by reorganizing only databases that need to be reorganized. The conditional reorganization feature eliminates unnecessary reorganizations and provides a history of resource savings.

Control block disassembly

The disassembly tool makes converting object code into source code quick and easy.

NOTE
The control block disassembly feature is available with all MAXM Reorg for IMS solutions.
DBD mapping

The DBD mapping feature lets you view, print, and export the segment structure of a database.

Exception management

MAXM Reorg/Online and MAXM Reorg/EP with Online/Defrag Feature Utilities lets you communicate with your system on an exception-only basis. The exception list displays the problems that have been identified by analysis.

JCL generator

The JCL generator is a DBA tool that generates JCL that is optimized for your environment.

**NOTE**
The JCL generator feature is available with all MAXM Reorg for IMS solutions.

Repository management

MAXM Reorg/Online and MAXM Reorg/EP with Online/Defrag Feature Utilities lets you define schedules to back up repositories and purge statistical data.

**NOTE**
The repository management feature is available with all MAXM Reorg for IMS solutions.
Database Management Console

MAXM Reorg/Online and MAXM Reorg/EP with Online/Defrag Feature Utilities is operated through a Windows user interface called the Database Management Console.

**NOTE**
The Windows user interface (Database Management Console) feature is available with all MAXM Reorg for IMS solutions.

Statistical information

MAXM Reorg/Online and MAXM Reorg/EP with Online/Defrag Feature Utilities offers interactive charts and tables that display current state and historical information about jobs, databases, database partitions, data sets, segments, and conditional reorganizations.

**NOTE**
The statistical information feature is available with all MAXM Reorg for IMS solutions.

MAXM Reorg/Online processing phases

Table 2 describes what happens to the original database and the shadow database during each phase of a MAXM Reorg/Online reorganization. MAXM Reorg/Online completes all phases in one job step.

<table>
<thead>
<tr>
<th>Phase</th>
<th>MAXM Reorg/Online action</th>
<th>Original (input) database</th>
<th>Shadow (output) database</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Delete/define shadow databases. Online Reorg function</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>2</td>
<td>Issue /BMCPAU. Enable recording of updates. Issue /STA.</td>
<td>No access</td>
<td>None</td>
</tr>
</tbody>
</table>
If an error occurs during reorganization, you can stop and start again later. Because the original data sets remain intact, you can abandon the reorganization and return to a prereorganization status of the database. This eliminates the need to take an image copy of the disorganized database before the reorganization.

Figure 2 on page 66 shows a MAXM Reorg/Online reorganization.

<table>
<thead>
<tr>
<th>Phase</th>
<th>MAXM Reorg/Online action</th>
<th>Original (input) database</th>
<th>Shadow (output) database</th>
</tr>
</thead>
</table>
| 3     | Reorganize and simultaneously take an image copy of the database one segment at a time by using the following functions:  
- Reorg function  
- Unload function  
- Reload function  
- Index Build function  
- Image Copy function  
- Prefix Resolution and Update function  
Capture the updates made to the database. | The Online Reorg function records the updates made while the database is being reorganized. | A reorganized shadow database is created, one segment at a time. |
| 4     | Apply the updates to the shadow database.                                                | The Online Reorg function continues to record updates while applying previous updates to the shadow. | DL/I batch program applies updates to the shadow (several iterations).                   |
| 6     | Recover the logs.                                                                       | Recording is off. The database is available.                                             | Correct the batch logs.                                                                 |
Online Reorg function captures changes to the database from online IMS/TM or CICS DBCTL and writes changes to logs.

Online Reorg function invokes Reorg function to reorganize shadow database. Online Reorg function continues capturing updates to online database.

Reorg function swaps the data set names, completes all DBRC notifications, and makes the database available for users.
Phase 1: Set up the database for reorganization

During phase 1, MAXM Reorg/Online uses an automated operator interface (AOI) to issue an IMS /DBR DB dbdName NOFEOV command to clear all buffers and stop the database. For more information, see “Automated operator interface” on page 126.
SNAPSHOT(N)

The AOI issues a /STA command to put the database back online. The Online Reorg function copies the database to shadow data sets and captures the IMS log records for the database and its primary index. When the database is copied, the Online Reorg function applies the logs to the copy so that the shadow database is identical to the original database.

The AOI issues a /DBR DB dbdName NOFEOV command to clear all buffers and stop the database. The database is started again.

SNAPSHOT(Y)

If you have EXTENDED BUFFER MANAGER for IMS (XBM for IMS) or SNAPSHOT UPGRADE FEATURE for IMS (SUF for IMS) enabled, the Online Reorg function invokes Snapshot Copy. Snapshot Copy preserves a point-in-time image of the database, eliminating a full database copy and the need to unload to temporary data sets, saving DASD and the overhead associated with taking a copy. It also reduces the elapsed time for the reorganization. (For more information, see “Snapshot Copy function” on page 178.) The Online Reorg function activates the recording mode, and issues a /STA command to put the database back on line. At this point, the database is reopened, DBRC authorization is granted, the Online Reorg function prepares to record the updates, and, using the point-in-time image of the database, the reorganization begins. The entire process should take only a few seconds.

Phase 2: Reorganize the database

During phase 2, Reorg function reorganizes to shadow data sets.

SNAPSHOT(N)

If you do not use an unload BMP, the Unload function unloads the database and writes it to a temporary file. The temporary file is allocated to DASD and requires no JCL changes; however, you can save DASD space by specifying a DFSURGU2 DD statement to write the HD Unload file to tape. When the unload is complete, the Reload function reads the temporary file and reloads the records directly to the shadow database.
SNAPSHOT(Y)

If you have XBM for IMS or SUF for IMS enabled, the Unload function unloads the databases using the point-in-time image of the database and passes the unloaded segments to the Reload function. The Reload function reloads directly to the shadow database data sets.

If the database has secondary index records, the Reorg function invokes the Index Build function and passes the secondary index information to it for processing. The Index Build function creates new secondary indexes for the reorganized database. The Image Copy function takes an image copy for databases that do not have logical relationships.

Throughout phase 2, the Online Reorg function captures changes that are made to the original database and writes work file records required for the Prefix Resolution and Update function.

Phase 3: Prefix resolution and update

Phase 3 only occurs for databases with logical relationships. During phase 3, work file records from phase 2 are processed and prefix resolution and update is performed. Image copies are made of the databases. For restart and safety reasons, all DBRC notifications are deferred.

Phase 4: Roll forward the changes

During phase 4, the changes that were made to the database and recorded during the reorganization are applied to the shadow data sets in a batch DL/I process. The log file that is needed to make the changes available for recovery purposes is created during this process. The batch program runs until most changes have been applied.

Phase 5: Make the database available

During phase 5, the Online Reorg function uses its AOI to issue a /DBR DB dbdName command to stop the update process. The AOI takes the same actions in this phase as it did in phase 1.

After the successful /DBR, the Online Reorg function applies the final changes to the database. The shadow database is now logically identical to the actual database. The Online Reorg function renames the shadow database data sets (to the original database data set names) and notifies DBRC.
The Online Reorg function issues the NOTIFY.REORG and NOTIFY.IC and registers the logs that were created during the roll-forward process. The log records have wrong time stamps because the database change records (record types 50, 51, and 52) were created during phase 5. The Online Reorg function marks the logs in error. The logs undergo log recovery (phase 6) so that they can be used in any recovery or change accumulation process. The log recovery phase changes the time stamp in the log records to reflect a time within phase 5.

The Online Reorg function issues a /STA command to start the database and make it available to users.

### Phase 6: Correct the logs

During phase 6, the Online Reorg function changes the time stamps on the logs that were created during phase 4 to reflect a time in phase 5. The Online Reorg function registers the new logs.

Correcting the logs should take only a few minutes unless the database had a heavy volume of updates during the reorganization.

### System requirements

MAXM Reorg solutions have the following system requirements:

#### Table 3  System requirements (part 1 of 3)

<table>
<thead>
<tr>
<th>System or software</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware</td>
<td>MAXM Reorg solutions are designed to operate on any CPU that supports any IBM-supported version of MVS. The hardware requirements are the same as those for the IBM IMS product. MAXM Reorg solutions are written in assembler language within the constraints of the assembler for OS/VS2.</td>
</tr>
<tr>
<td>MVS</td>
<td>The Online Reorg function and Online/Defrag function require MVS 4.2.2 or later; the Cross-System Communications facility (XCF) and Name Services are required. MAXM Reorg solutions require MVS 4.1 or later. The products will support new versions of MVS and OS/390 as they become available from IBM. If you are using the BMC Cross-system Image Manager (XIM) feature available in Unload function and Reload function, MVS Version 4.3 or later is required.</td>
</tr>
</tbody>
</table>
### System requirements (part 2 of 3)

<table>
<thead>
<tr>
<th>System or software</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS</td>
<td>You must be a licensed user of the IMS/ESA Database Manager product. MAXM Reorg solutions support all IBM-supported versions of IMS and will support new versions as they become available from IBM. The Online/Defrag function supports IMS 5.1 (and later) and will support new versions as they become available from IBM. HALDBs require IMS 7 (and later).</td>
</tr>
<tr>
<td>DBRC</td>
<td>DBRC is required for the Online Reorg function and Online/Defrag function. DBRC must be active, and all databases must be registered. BMC recommends DBRC for MAXM Reorg solutions. Online/Defrag does not require that databases be registered unless you request dynamic allocation of the databases.</td>
</tr>
<tr>
<td>Sort product</td>
<td>MAXM Reorg solutions use any sort product that is compatible with the IBM OS/VS Sort/Merge Program. The required DD statements vary by product as follows:</td>
</tr>
<tr>
<td></td>
<td>■ CASORT requires the following DD statements, where ( x ) represents the task number:</td>
</tr>
<tr>
<td></td>
<td>— SORTCNTL</td>
</tr>
<tr>
<td></td>
<td>— SRT( x )CNTL</td>
</tr>
<tr>
<td></td>
<td>— SRA( x )CNTL</td>
</tr>
<tr>
<td></td>
<td>■ SYNCSORT requires the $SORTPARM DD statement.</td>
</tr>
<tr>
<td></td>
<td>■ DFSORT requires the DFSPARM DD statement.</td>
</tr>
<tr>
<td></td>
<td>Ensure that the environment set-up member (DLIGSET0) specifies ddnames for the SORTPRM keyword that are appropriate for your sort product. You can specify two ddnames for the SORTPRM keyword in the environment set-up member. If you need additional ddnames, you must specify them in the job’s JCL.</td>
</tr>
<tr>
<td>CICS</td>
<td>If you are using CICS with the Online Reorg function or Online/Defrag function, you must use the IMS DBCTL feature. Local DL/I is not supported.</td>
</tr>
<tr>
<td>Installation and maintenance</td>
<td>MAXM Reorg solutions are easy to install with IEBCOPY and some simple modifications to the logon procedures of users. You can install the utilities with the Quick Install System (QIS) or a manual installation procedure.</td>
</tr>
<tr>
<td></td>
<td>Because BMC provides complete replacements for all product modules, the procedure for installing maintenance is similar to the initial installation procedure. Installing product maintenance does not require a new CPU authorization password.</td>
</tr>
<tr>
<td></td>
<td>For more information, see the <em>BMC Products for IMS Installation Guide</em>.</td>
</tr>
</tbody>
</table>
This section describes limitations.

### All functions

MAXM Reorg solutions cannot be run under the Utility Control Facility (UCF).

### Online Reorg function

The following limitations exist in the Online Reorg function:

- The Online Reorg function does not support selected partition reorganizations for HALDBs with logical relationships.

- The following limitations apply to databases with logical relationships:
  - All LCHILD segments must be uniquely keyed.
  - All segments from the root segment down to and including the logical parent segment must be uniquely keyed.
  - Where virtual pairing is used, the delete rule for the real logical child must be V.
The Online Reorg function does not support CICS local DL/I.

DL/I batch programs that attempt to access the database being reorganized will not be able to obtain access. A DBRC “not authorized” message will be issued. If the Online Reorg function detects a DL/I batch program that is executing, the Online Reorg function terminates.

The following limitations apply to HISAM databases:

— All segments must have unique keys.
— For any secondary indexes of a HISAM database, the target must point to the root segment.

To use the AR/CTL interface, you must be using IMS 4.1 or later.

To use the Snapshot Copy function, you must have the snapshot technology level of XBM 4.4.00 or later.

DBD parameter changes during an Online Reorg function reorganization are limited. For more information, see “OLDDBD” on page 496.

The minimum VSAM share level must specify SHAREOPTIONS(2,3).

Any limitations in the Reorg function, the Unload function, the Reload function, the Index Build function, and the Image Copy function are limitations in the Online Reorg function.

Reorg function

The following limitations exist in the Reorg function:

— You cannot change certain characteristics of the DBD between the unload process and the load process. For more information, see “OLDDBD” on page 496.
— You cannot use the Reorg function to rebuild secondary indexes that are shared between different primary databases.
— The Reorg function accepts some, but not all, keywords from the other MAXM Reorg functions. For more information, see “MAXM Reorg keywords and the REORG command” on page 575 and “Keywords available with the REORG command” on page 576.
— Any limitations for the Unload function, the Reload function, and the Index Build function are limitations in the Reorg function.
Unload function

The following limitations exist in the Unload function:

- Parallel unloading of the database when root keys are compressed, regardless of database organization, is not supported. The Unload function does support HDAM or PHDAM root segments with compressed keys for all other functions and features.

- HSAM and SHSAM databases are not supported.

- In an HDAM or PHDAM database, all root segments must have a sequence (key) field. The sequence field does not have to be unique.

- Secondary index databases are not supported as stand-alone databases. Use the Index Build function to process secondary index databases.

- The target segment of a PROCSEQ PCB must be the root segment.

Reload function

The following limitations exist in the Reload function:

- For HDAM and PHDAM DBDs, you must specify a maximum relative block number (RBN) in the RMNAME operand of the DBD macro.

- In an HDAM or PHDAM database, all root segments must have a sequence (key) field. The sequence field does not have to be unique.

- The Reload function does not provide direct support for the HISAM unload and reload utilities (DFSURUL0 and DFSURRL0). The Reload function can reload the HD Unload data set (DFSURGU1) of a HISAM database created by DFSURGU0. The Reload function does not support HISAM databases with multiple data set groups.

- You must invoke the HD Sort utility (before or during the RELOAD job step) when you are performing any of the following tasks:
  - change the key of a root segment
  - convert an HDAM database to HIDAM or HISAM
  - convert a PHDAM database to a PHIDAM
  - convert a HIDAM or HISAM database to HDAM
  - convert a PHIDAM database to PHDAM
  - change the randomizer parameters for an HDAM or PHDAM database (except maximum insert bytes)
Limitations

**Index Build function**

The following limitations exist in the Index Build function:

- You can use the Index Build function to rebuild an *entire* shared secondary index; however, the Index Build function does not support the partial loading of a shared secondary index. The Index Build function performs all index functions preceding the actual loading of a specific secondary index into the shared index database.

- Output shared secondary index data sets must be empty.

**Prefix Resolution/Update function**

The following limitations exist in the Prefix Resolution and Update function:

- Execute the IMS region controller (DFSRRC00) only when executing the following programs:
  - user database load programs
  - BMC or IMS database reorganization reload utilities (DFSURGL0)
  - Prefix Update (DFSURGP0), unless you are using PRPURGP0 or the Concurrent Prefix Update function

  The Prefix Resolution function (DFSURG10) cannot be run under the region controller.

- The Work File Generator cannot be used when building multiple databases in one job step.

- Because the Prefix Update function (PRPURGP0) does not use the IMS buffer handler to update databases, logging is not supported. No IEFRDER DD statement is required.

- The Prefix Update function cannot process the DFSURWF3 data set that is created by the IMS Prefix Resolution utility because the relative byte addresses (RBAs) are not always in physical sequence.

- The Prefix Update function supports HDAM and HIDAM databases and supports OSAM and VSAM access methods; however, it does not support HISAM databases of any kind. If you have logical relationships in HISAM databases, split the output of the Logical Twin Resolution (PRPURG30) phase so that you can run the IMS Prefix Update utility for the HISAM databases and the Prefix Update function for the HDAM and HIDAM database.
The Prefix Update function does not update logical parent (LP) counter fields to zero. These fields are updated only when you delete a segment type involved in a logical relationship. In this case, specify FORCELP(Y) in the PLUSIN control statement data set during Prefix Resolution. FORCELP(Y) is available only through the PLUSIN control statement.

**NOTE**
The FORCELP(Y) option degrades performance and is valid only for a DBIL-type reorganization.

When running the Prefix Update function for a multi-volume OSAM database, you must first run the BMC-provided program DBUVTOC. For information about this program, see the BMC IMS Database Supplemental Utilities Reference Manual.

**Image Copy function**

The following considerations and limitations apply to the use of the Image Copy function:

- If you need to recover a database that was reorganized by the Online Reorg function, note that the image copy time stamp in the header of the image copy data set created by the Online Reorg function is not the same as what is posted in DBRC. The time stamp in the image copy header is created at image copy initialization, and the NOTIFY.IC is posted at the end of the reorganization for data integrity purposes. You should be aware of this situation because it can affect recovery.

RECOVERY PLUS for IMS handles this situation with no manual intervention. However, if you are using the IMS Recovery utility, the utility will issue a message indicating that the time stamps do not match. To avoid zapping the IMS Recovery utility to bypass this check, BMC recommends that you concatenate the load library in your IMS Recovery utility STEPLIB concatenation (ahead of RESLIB) when you recover a database using the image copy created during the Online Reorg function reorganization. Including the load library in your IMS Recovery utility STEPLIB concatenation ahead of RESLIB causes the Online Reorg function to dynamically bypass checking the time stamps (no USERMOD or zap is required).

- If a database consists of multiple data set groups or has indexes, the image copies cannot be stacked.

- You cannot create a compressed image copy by using the Image Copy function.
If any indexes are set to NONRECOV in DBRC, they will not be image copied and a NOTIFY.UIC will be issued. DBRC must be active for index databases to be image copied.

If a secondary index is set to RECOV in DBRC, the Image Copy function attempts to take an image copy. If a secondary index is recoverable and an image copy data set cannot be allocated, either dynamically or through JCL statements, the Image Copy function fails when it attempts to take an image copy of the index.

You can use dynamic allocation models and matrixes when you run the Image Copy function under a reorganization function.

**Online/Defrag function**

The Online/Defrag function has the following limitations:

- HDAM, HIDAM, PHDAM, and PHIDAM databases are supported. No other database types are supported.

- Databases that have any type of logical relationship are not supported.
Reorg function (Fast Reorg Facility/EP)

This chapter describes how to use the Reorg function to reorganize a database in one job step. This chapter includes the following topics:

- Overview ................................................. 80
- Requirements ........................................... 82
- JCL requirements ...................................... 83
  - Standard Reorg function reorganization ........ 83
  - Reorganization using an HD Unload file as input 84
  - Reorganization for logically related databases 86
  - Reorganization to an image copy ................. 91
  - Reorganizing multiple databases .......... 92
  - DD statements ..................................... 92
  - PLUSIN examples .................................. 93
  - JCL considerations ................................ 94
  - Unused files ....................................... 97
- Keywords ............................................... 97
- Global options module ............................... 101
- Reports ............................................... 102
- Post-processing phase ............................... 103
- Empty database initialization .................... 103
- Operational considerations ......................... 104
  - DBRC authorization .............................. 104
  - Secondary indexes ............................... 105
  - Image copies ..................................... 106
  - Database replication ............................ 108
Overview

With the Reorg function, the input (disorganized) database and the output (reorganized) database use different data sets. A segment is unloaded by the Unload function and is passed directly to the Reload function for reload unless you specify the OLDDBD keyword. For more information, see “OLDDBD” on page 496.

The index maintenance records, if any, are passed to the Index Build function. This process continues for each segment. The Index Build function sorts the index records and creates the index databases. You can take an image copy of the database as it is reorganized.

The Reorg function automatically invokes the prefix resolution and update function if the database has logical relationships. (For limitations, see page 73.)

The term standard reorganization refers to a reorganization where you use the database, rather than an HD Unload file, as input. To execute a standard Reorg function reorganization, specify INPUT(DB) or accept it as the default.

Figure 4 shows the system flow for the Reorg function.

**Figure 4  Reorg function system flow**

Figure 5 on page 81 shows the input and output processes of a standard Reorg function reorganization.
Table 4 lists the names that are recognized by the Reorg function in a standard reorganization. It shows the relationship between input names (from the disorganized database) and output names (for the reorganized database).

<table>
<thead>
<tr>
<th>Names</th>
<th>Input (to the Unload function)</th>
<th>Output (from the Reload function)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ddnames</td>
<td>ddnames defined with the UNLDD or UNLDDLST keywords or dynamically allocated</td>
<td>DBD ddnames</td>
</tr>
<tr>
<td>data set names</td>
<td>disorganized primary index and database data set names (the names registered to DBRC)</td>
<td>reorganized primary index and database data set names</td>
</tr>
<tr>
<td></td>
<td>No secondary indexes are used as input.</td>
<td>old or new secondary index data sets:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Old data set names are excluded from swapping, and you cannot have read access during reorganization.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ New data set names are swapped, and you can have read access during reorganization.</td>
</tr>
</tbody>
</table>
Reorganizing a database with the Reorg function requires the following actions:

**NOTE**
The Reorg function supports HALDB reorganizations. The JCL and options are the same for full-function databases and HALDBs.

- **Required.** Install the Reorg function (with the prefix resolution and update function) and the versions of the Unload function, Reload function, and Index Build function included on the same tape. If you plan to use the Parallel Image Copy feature, you must install IMAGE COPY PLUS and the BMC load library from the same tape. If you want IMAGE COPY PLUS to invoke POINTER CHECKER PLUS to check pointers during the image copy, install POINTER CHECKER PLUS from the same tape. For instructions, see the *BMC Products for IMS Installation Guide*.

- **Required.** Prepare the databases for reorganization. Each database must be offline or in read-only mode. You can use the AOI feature of the Online Reorg function to accomplish this automatically.

  If you are not using the AOI, issue a `/STA DB dbdName ACCESS=RD` command for each database to change the DBRC access intent.

  Issue one of the following commands for each database that is to be reorganized:

  — If you specify SHARE(YES) to allow read access during the reorganization, issue a `/DBD dbdName NOFEOV` command for each database.

  — If you specify SHARE(NO), issue a `/DBR dbdName NOFEOV` command for each database.

  Issue the `/DBD` or `/DBR` commands in a separate step before the reorganization job step. If you specify AUTHERR(WTOR) in the Reorg function job step and you have not issued `/DBD` or `/DBR` commands for all databases or you did not use the AOI, the Reorg function issues messages stating which databases must be stopped.

- **Required.** Create and submit one set of JCL that runs the entire reorganization process. You can use your existing LOADPLUS, FRF, or MAXM Reorg classic JCL and add a few statements specific to the Reorg function. You also need to add IDCAMS statements to allocate the output data sets if you did not specify the IDCAMS keyword. For details, see “JCL requirements” on page 83. The PLUSIN DD statement and the REORG command are required.

- **Optional.** If you want any special processing, set keywords in the PLUSIN DD statement. For an overview of the keywords, see Table 5 on page 97.
JCL requirements

The JCL requirements vary, depending on several factors. This section describes the JCL requirements for the following scenarios:

- reorganizing a database (standard Reorg function reorganization)
- reorganizing a database by using an HD Unload file created in a separate step before the reorganization as input to the reorganization
- reorganizing a database that has logical relationships
- reorganizing a database to an image copy

Standard Reorg function reorganization

The JCL shown in Figure 6 shows two steps involved in reorganizing an HDAM or PHDAM database using shadow data sets. The first step allocates the shadow data set, and the second step reorganizes the database. The PLUSIN control statement shows the REORG command and some optional keywords. Use this sample JCL for databases that do not have logical relationships.

Figure 6  Reorg function JCL (part 1 of 2)

```bash
//ALLOC EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
DELETE your.shadow.database.dataSet
SET MAXCC=0
DEFINE CLUSTER(NAME(your.shadow.database.dataSet) -
........) -
```
Reorganization using an HD Unload file as input

The JCL shown in Figure 7 shows how to reorganize a database by using an HD Unload file as input. You must create the HD Unload file in a separate step before beginning the Reorg function reorganization. This example shows a non-parallel unload and reload. (To reload a database in parallel mode, you must have unloaded the database in parallel mode by using the Unload function.)

**NOTE**

For best performance, BMC recommends that you do not use this method of reorganizing databases. Use a standard Reorg function reorganization (INPUT(DB)) instead.

---

### Figure 6  Reorg function JCL (part 2 of 2)

```plaintext
DATA(NAME(your.shadow.database.dataSet.DATA))
//REORG EXEC PGM=DFSRRC00,PARM='ULU,DFSURGLO,dbName'
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//AMSOUT DD DISP=SHR,DSN=your.restart.info.dataSet
//BMCMSG DD SYSOUT=* 
//BMCPRINT DD SYSOUT=* 
//PLUSIN DD *
REORG SWAP(YES) UNLDD(OLD,1) ICP(YES) INPUT(DB)
//ICPSYSIN DD *
GLBL PCP(Y)
AIC DBD(dbName) DDN(dbdddnam) IC(imagecpy)
//imagecpy DD DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
SPACE=(CYL,(60,10),RLSE),DSN=image.copy.dataSet
//dbdddnam DD DISP=SHR,DSN=original.database.dataSet
//OLDddnam DD DISP=SHR,DSN=original.database.dataSet
```

For details about IDCAMS commands and allocation of output data sets, see the IBM reference.

For more information about the DD statements, see “Reorganizing multiple databases” on page 92 and Chapter 13, “JCL requirements.”

### Figure 7  Reorg function JCL when using HD Unload file as input (part 1 of 2)

```plaintext
//UNLOAD EXEC PGM=DFSRRC00,PARM='ULU,DFSURGLO,dbName'
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//DFSURGUD DD DISP=OLD,DSN=test.hdunload
//dbdddnam DD DISP=SHR,DSN=original.database.dataSet
//BMCMSG DD SYSOUT=* 
```

---
Reorganization using an HD Unload file as input

Chapter 2 Reorg function (Fast Reorg Facility/EP) 85

For more information about DD statements, see “Reorganizing multiple databases” on page 92. The following DD statements are specific to the Reorg function REORG command when you specify INPUT(UNLDFILE):

**DFSUINPT DD**

Required if you specify INPUT(UNLDFILE). Specify a DFSUINPT DD statement if the database was unloaded in non-parallel mode. (See “DFSUINPT DD” on page 358.) Specify the appropriate number of DFSUINnn DD statements if the database was unloaded in parallel mode. (See “DFSUINnn DD” on page 358.)

**dbdddnam DD**

Required. Defines the data sets containing the reloaded database. One statement of this type must be present for each data set appearing in the DBD that describes the database. The ddnames must match the ddnames in the DBD. All primary database data sets, secondary index data sets, and primary index data sets must be included in the JCL.

For more information, see “Multivolume data sets” on page 225.
Reorganization for logically related databases

Logical relationships provide a user-defined access path to data. Logical relationships are typically established by a special set of pointers: logical parent (LP) and logical child (LC).

Logical pointers can be symbolic or direct. A symbolic pointer uses the logical parent concatenated key (LPCK) to point to the logical parent segment. A direct pointer uses the RBA to point to the logical parent or logical child segment.

When a database is reorganized, all segments in the database are often moved to different locations. Consequently, all direct pointer references to the segments must be updated during the reorganization. Symbolic pointer references do not require an update because they use the LPCK.

The logical relationship is either internal or external.

- Internal—LP and LC are in the same database.
- External—LP and LC are in different databases.

**NOTE**

For a comprehensive explanation of logical relationships, refer to the IBM IMS/ESA Administration Guide: Database Manager.

Figure 8 on page 87 depicts three types of logical relationships.

- Internal logical relationship
- External logical relationship with symbolic pointers only
- External logical relationship with direct pointers

The internal logical relationship is between two segments in the same database D1. The logical child segment D1S3 points to the logical parent segment D1S2 using the LPCK of the logical parent. The LPCK of the logical parent is stored in the data portion of the logical child. The logical parent does not point back to the logical child.

The external logical relationship with symbolic pointers only is between a logical child segment D1S5 in D1 and a logical parent segment D2S1 in D2. The nature of the relationship is the same as the internal logical relationship only the segments reside in different databases.

If the database data set is a VSAM data set, you can use the AMP parameter. For more information, see the discussion of VSAM KSDS buffers on page 186.
The external logical relationship with direct pointers is a bidirectional relationship between a logical child segment D1S4 in D1 and a logical parent segment D2S3 in D2. The logical child points to the logical parent using the LPCK of the logical parent. The logical parent points back to the logical child using the RBA of the logical child. The RBA of the logical child D1S4 is stored in the prefix of D2S3.

**Figure 8  Logical relationships**

![Diagram showing logical relationships between segments D1 and D2]

**Single-step reorganization**

The single-step reorganization incorporates the prefix resolution and update process into the Reorg function. The Online Reorg function and Reorg function support databases with internal and external logical relationships.

The Online Reorg function always runs the prefix resolution and update process internally. If DFSURWF1 is present, prefix resolution and update is deferred. If neither work files are present, prefix resolution and update occurs internally.

**NOTE**

If you are reorganizing a database that is the target of direct pointers from an external database, you must also reorganize the external database.
Figure 9 shows sample JCL for a single-step reorganization. Use the JCL Generator in MAXM Database Advisor for IMS to generate JCL that is optimized for your environment. The JCL performs the following tasks:

- creates a control data set (DFSURCDS)
- allocates output data sets
- reorganizes the database including prefix resolution and update
- automatically swaps the data sets

**Figure 9 Reorg function JCL for single-step reorganization**

```plaintext
//STEP1 EXEC PGM=DFSRRC00,PARM='ULU,DFSURPRO'
//STEPLIB DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//SYSPRINT DD SYSOUT=* 
//DFSURCDS DD DISP=(,CATLG),DSN=dfsurcds,UNIT=SYSDA, DCB=BLKSIZE=1600,SPACE=(CYL,(1,1))
//SYSIN DD *
DBIL=dbname
//ALLOC EXEC PGM=IDCAMS
//SYSPRINT DD SYSPRINT
//SYSPRINT DD *
DELETE frf.shadow.database.dataset
SET MAXCC=0
DEFINE CLUSTER(NAME(frf.shadow.database.dataset) - .......) -
DATA(NAME(frf.shadow.database.dataset.DATA))
//REORG EXEC PGM=DFSRRC00,PARM='ULU,DFSURGLO,dbname'
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
//IMS DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//AMSOUT DD DISP=SHR,DSN=frf.restart.info.dataSet
//BMCMSG DD SYSPRINT
//BMCPRINT DD SYSPRINT
//PLUSIN DD *
REORG SWAP(YES) UNLDD(OLD,1) ICP(YES)
//DFSURCDS DD DISP=OLD,DSN=dfsurcds
//dbddnam DD DISP=SHR,DSN=frf.shadow.database.dataset
//OLDddnam DD DISP=SHR,DSN=original.database.dataset
/*
```

**Multi-step reorganization**

If you do not want the Reorg function to automatically perform prefix resolution and update or build secondary indexes, add the DFSURWF1 card (and optionally the PRPURWF1 card) to the reorganization step.

You must also add the appropriate utility steps to the JCL to perform prefix resolution and update. Parameter options ICP(YES) and SWAP(YES) may not be specified.
For examples of keywords used in the PLUSIN control statement data set, see “PLUSIN examples” on page 93.

Figure 10 shows sample JCL for the multi-step reorganization. The JCL performs the following tasks:

- creates a control data set (DFSURCDS)
- allocates output data sets
- reorganizes the database
- invokes the prefix resolution and update function
- invokes the RESTART process (to swap data set names)

**Figure 10  Reorg function for JCL for multi-step reorganizations (part 1 of 2)**

```
//PREREORG  EXEC PGM=DFSRRC00,PARM='ULU,DFSURPR0'
//STEPLIB   DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB  DD DISP=SHR,DSN=IMSVS.RESLIB
///IMS      DD DISP=SHR,DSN=IMSVS.DBDLIB
//SYSPRINT DD SYSPRINT DD SYSOUT=* 
//DFSURCDS DD DISP=(,CATLG),DSN=test.dfsurcds,UNIT=SYSDA,
//            DBC=BLKSIZE=1600,SPACE=(CYL,(1,1))
//SYSIN     DD * 
//DBI=dbdName
//**=======================================================================
//ALLOC EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=* 
//SYSIN    DD *
//DELETE  frf.shadow.database.dataSet
//SET MAXCC=0 
//DEFINE CLUSTER(NAME(frf.shadow.database.dataSet) -
//........) -
//DATA(NAME(frf.shadow.database.dataSet.DATA))

//******************************
//REORG STEP - FRF/EP
//******************************
//REORG EXEC PGM=DFSRRC00,PARM='ULU,DFSRGL0,dbdName'
//STEPLIB   DD DISP=SHR,DSN=bmc.xxx.load
//         DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB  DD DISP=SHR,DSN=IMSVS.RESLIB
///IMS      DD DISP=SHR,DSN=IMSVS.DBDLIB
//AMSOUT    DD DISP=SHR,DSN=frf.restart.info.dataSet
//BMCMSG    DD SYSPRINT DD SYSOUT=* 
//BMCPRINT  DD SYSPRINT DD SYSOUT=* 
//PLUSIN    DD * 
//REORG SWAP(NO) UNLDD(OLD,1) ICP(NO)
//DFSURCDS  DD DISP=OLD,DSN=test.dfsurcds
//DFSURWF1  DD DISP=(NEW,CATLG,DELETE),SPACE=(TRK,(1,1)),
//            DSN=test.dfsurwf1,UNIT=SYSDA
//PRPURWF1  DD DISP=(NEW,CATLG,DELETE),SPACE=(CYL,(10,10)),
//            DSN=test.prpurwf1,UNIT=SYSDA
//OLDddnam  DD DISP=SHR,DSN=original.database.dataSet
/*
```
**Figure 10**  Reorg function for JCL for multi-step reorganizations (part 2 of 2)

```plaintext
//***************************************************************
//**          PREFIX RESOLUTION STEP                           
//***************************************************************
/*
//PREFRES EXEC PGM=DFSURG10
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//SYSPRINT DD SYSOUT=*,DCB=BLKSIZE=1210
//SYSOUT DD SYSOUT=* 
//PLUSOUT DD SYSOUT=* 
//PLUSIN DD *
GLBL SORTIN(N) UPDATE(Y) HIPO(Y)
/*
//PLUSLIST DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB 
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(5,3),RLSE)
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(5,3),RLSE)
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(5,3),RLSE)
//DFSURWF3 DD DISP=(,PASS),DSN=&&WF3,UNIT=SYSDA,SPACE=(CYL,(100,50))
//DFSURCSD DD DISP=SHR,DSN=test.dfsurcsd
//SORTIN DD DISP=SHR,DSN=test.ppurrwfl
//dbdName DD DISP=SHR,DSN=frf.shadow.database.dataSet
Initializing required libraries and DD statements.

//***************************************************************
//**    **** FRF/EP RESTART  -  DOES THE IDCAMS ALTER COMMANDS 
//**         AND THE DBRC COMMANDS THAT WERE POSTPONED IN PREVIOUS 
//**         FRF/EP STEPS.       (SWAP=N WAS SPECIFIED EARLIER BECAUSE 
//**                              OF LOGICAL RELATIONSHIPS) 
//***************************************************************
/*
//RESTART EXEC PGM=DFSRRC00 
//PARM='ULU,DFSURG10,dbdName'
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB 
//PLUSIN DD *
RESTART
/*
//AMSOUT DD DISP=SHR,DSN=frf.restart.info.dataSet
//BMCTRACE DD SYSOUT=* 
//SYSOUT DD SYSOUT=* 
//BMCPRTF DD SYSOUT=* 
//BMCMMSG DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 
/*
```
Reorganization to an image copy

You can save DASD space by specifying SWAP(x,IC). When you specify SWAP(x,IC), the segments being reloaded are written directly to the image copy data sets and not to the shadow data sets. When all of the reorganized segments are written to the image copy, the image copy data sets are restored to the original database data sets. Because the secondary indexes are rebuilt directly to the original database data set names, the image copies of the secondary index data sets (if any) are not restored to the original secondary index data sets. If the database contains logical relationships such that prefix resolution and update is required, SWAP(x,IC) is not supported.

The JCL shown in Figure 11 shows the steps involved in reorganizing a HIDAM database (with no secondary indexes) to an image copy. Because you are reorganizing to an image copy rather than shadow data sets, you do not need to DELETE/DEFINE any data sets. The PLUSIN control statement shows the REORG command, the SWAP(Y,IC) and ICP(YES) keywords, and some optional keywords. Use this sample JCL for databases that do not have logical relationships.

For more information about reorganizing to an image copy, see “SWAP” on page 548.

---

**Figure 11  Reorg function JCL for SWAP(x,IC)**

```plaintext
//FRFEP    EXEC PGM=DFSRRC00,REGION=0M,
//         PARM='ULU,DFSURGL0,HIDAMVSM,,,,,,,,,,,Y'
//STEPLIB  DD  DISP=SHR,DSN=bmc.xxx.load
//         DD  DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD  DISP=SHR,DSN=IMSVS.RESLIB
//IMS      DD  DISP=SHR,DSN=IMSVS.DBDLIB
//AMSOUT  DD  DISP=OLD,DSN=your.restart.info.dataSet
//PDX      DD  DISP=SHR,DSN=your.bmcdbu.pdx
//PLUSIN   DD  *
//REORG SWAP(Y,IC) ICP(Y) UNLDD(OLD,1) SHARE(Y)
/*
//ICPSYSIN DD  *
GLBL DBRC(Y)
IC DBD(HIDAMVSM) DDN(HIDAMVSM) IC(IC1)
IC DBD(PRIMENDX) DDN(PRIMENDX) IC(IC2)
/*
//IC1      DD  DISP=(,CATLG,DELETE),DSN=your.hidamvsm.ic.dataSet,
        UNIT=SYSDA,SPACE=(CYL,(60,10),RLSE)
//IC2      DD  DISP=(,CATLG,DELETE),DSN=your.primendx.ic.dataSet,
        UNIT=SYSDA,SPACE=(CYL,(60,10),RLSE)
/*
//OLDAMVSM DD  DISP=SHR,DSN=your.original.hidamvsm.db
//OLDMENDX DD  DISP=SHR,DSN=your.original.primendx.db
/*
//BMCPRINT DD  SYSOUT=* 
//BMCMSG   DD  SYSOUT=* 
//BMCTRACE DD  SYSOUT=* 
//SYSOUT   DD  SYSOUT=* 
//SYSPRINT DD  SYSOUT=* 
//SYSDUMP  DD  SYSOUT=* 
/*
```
Reorganizing multiple databases

BMC supports reorganizing (and copying) multiple databases in one job step. For detailed information about this feature, refer to Chapter 5, “Multiple databases.”

DD statements

The following DD statements are specific to the Reorg function REORG command:

**dbdddnam DD**

Required for secondary indexes. Describes the data set that will contain the secondary index databases after they are reorganized (the original database).

If you specify the DYNALLOC keyword to dynamically allocate secondary index database data sets, see “DYNALLOC” on page 430.

If you specify SWAP(x,IC) to reorganize the database to an image copy data set, see “SWAP” on page 548.

If you do not use the DYNALLOC keyword to dynamically allocate the secondary index database data sets, you must meet the following conditions:

- The secondary index data sets and ddnames must use the names that are registered to DBRC.
- Specify the primary database and primary index on the OLDddnam DD statements.
- Do not specify the dbdddnam DD statement for the primary database and primary index database.

**OLDddnam DD**

Required. Describes the data set containing the primary database and primary index (if applicable) being reorganized. Do not specify DD statements for the original (disorganized) secondary index data sets; the Reorg function ignores these DD statements. For more information, see “UNLDD” on page 560 and “UNLDDLST” on page 562.

If you specify the DYNALLOC keyword to dynamically allocate database data sets, see “DYNALLOC” on page 430.

If you specify SWAP(x,IC) to reorganize the database to an image copy data set, see “SWAP” on page 548.
Whether you have the Reorg function swap the data set names or you swap them manually, the original (disorganized) data sets and the reorganized (shadow) data sets must be in the same MVS catalog. After the data sets are reorganized, they are renamed to have the original data set names. If the data sets are in different MVS catalogs, MVS would not be able to find the renamed data sets. The best way to ensure that the data sets are in the same catalog is to use the same high-level qualifier for both sets of data set names.

You can have the original and reorganized data sets in different MVS catalogs if you want to replicate a database. For more information, see “Database replication” on page 108.

**olddbd DD**

Required if you specify the OLDDBD keyword. Describes the data set containing the original DBD (before you make any changes). Use the IMS DD statement to specify the new DBD.

**ICPSYSIN DD**

Optional. Contains the commands and keywords for the Image Copy function. The IC or AIC command is required for each database that is being reorganized and is registered as RECOV in DBRC. If the ICPSYSIN DD statement is omitted the Online Reorg function will automatically create an IMAGE COPY for all databases being reorganized that are registered as RECOV in DBRC. For more information, see “JCL requirements” on page 275.

**imagecpy DD**

Optional. You can code Image Copy function keywords to dynamically allocate this data set. Describes the data set containing the image copy. You must have one imagecpy DD statement for each database being reorganized. You cannot stack image copy data sets. For more information, see “JCL requirements” on page 275.

**PLUSIN examples**

The following examples show some possible combinations of keywords:

**Example 1:**
The Reorg function swaps the data set names automatically after the reorganization is complete:

```
REORG SWAP(YES)
```
**Example 2:**
Reorg function reorganization of database with internal logical relationships.

```
REORG SWAP(YES) ICP(YES)
```

**Example 3:**
The database remains available for read access during the reorganization. The data set names are swapped automatically. The Image Copy function is invoked to take an image copy:

```
REORG SHARE(YES) SWAP(YES) ICP(YES)
```

**Example 4:**
The input and output database data sets are reorganized and dynamically allocated. The reorganized data sets are appended with .K:

```
REORG DYNALLOC(YES,YES,K) UNLDD(OLD,1)
```

**Example 5:**
The database is reorganized, and the files IDCIN and IDCOUT are used for IDCAMS processing:

```
REORG IDCAMS(IDCIN,IDCOUT)
```

## JCL considerations

The Reorg function uses database data sets and ddnames in specific ways, depending on the portion of the database that the data set represents and the database type. The following guidelines will help you build your JCL correctly.

### ddnames

For all primary database data sets, regardless of the value of the SWAP keyword, the ddnames that are defined by the UNLDD or UNLDDLST keyword point to the disorganized (original) input data sets that can be registered to DBRC. The ddnames that are defined by the UNLDD or UNLDDLST keyword are the substituted DBD ddnames, not the original DBD ddnames. If you specify SWAP(YES,IC), VSAM data sets must be defined with the REUSE option.

If you specify SWAP(x,IC), the ddnames that are defined in the DBD are not used because no shadow data sets are used. However, the substituted DBD ddnames are used for the original database data sets.
If you specify INPUT(UNLDFILE), the ddnames that are defined in the DBD point to the original data sets that are registered to DBRC. No shadow data sets exist.

--- WARNING ---

If you are reorganizing a database with the Online Reorg function, do not specify the substituted ddnames with the original, DBRC-registered data sets. Specify only the DBD ddnames with the shadow data set names.

--- Secondary indexes ---

For all secondary index files, the following statements are always true:

- Secondary index files must be output files only and must be empty unless the VSAM REUSE option is used.

--- NOTE ---

The Reorg function honors the VSAM REUSE option for all data sets except shared secondary index data sets. Shared secondary index data sets must be empty.

- The secondary index ddnames are always those defined in the DBD.

- The data set names depend upon the value of the SHARE keyword:
  - If you specify SHARE(YES), the secondary index output data sets cannot be the data sets that are already registered to DBRC.
  - If you specify SHARE(NO), the output data sets can be new data set names or the data set names that are being used for the secondary indexes.

If you specify SWAP(YES) and SHARE(NO) and the secondary indexes have been created by reusing the current files, the secondary index data sets are excluded from swapping. All other data sets are swapped. Fall-back to prereorganization status of the database would require that these secondary indexes be rebuilt or recovered.

--- ddnames and SWAP ---

In a standard Reorg function reorganization, if you specify any value for the SWAP keyword other than SWAP(x,IC), the following requirements exist:

- HDAM and PHDAM databases always require at least two unique ddnames and two unique data set names (one each for input and output).
HIDAM and PHIDAM databases always require at least four ddnames and four data set names (one input and one output each for the primary database and the primary index).

If the database contains multiple data set groups, each data set group requires two ddnames and two data set names (one each for input and output) for each data set group.

**DFSURWF1**

Without the Reorg function, the DFSURWF1 file is required to contain the logical relationship and secondary index data. The Reorg function does not write the secondary index records to DFSURWF1, as the secondary index build is integrated into the Reorg function execution.

The following considerations apply to the DFSURWF1 file:

- With the Reorg function, the DFSURWF1 and PRPURWF1 files are used only for logical relationships. If the database to be reorganized does not include logical relationships, you can omit DFSURWF1.

- The DFSURWF1 and PRPURWF1 files can be omitted from the JCL to instruct the Reorg function to perform integrated prefix resolution and update.

- When the Reorg function is executed with prefix resolution and update integrated, DFSURWF1 as DUMMY indicates that secondary indexes should be initialized with zero records.

- If prefix resolution and update are to be run as separate JCL steps, you must specify a DFSURWF1 file, PRPURWF1 file, or both for collecting the prefix update records.

- If you want the Reorg function to collect the prefix resolution and update records in IBM utility format, include a DFSURWF1 file, and omit PRPURWF1 or specify PRPURWF1 as DUMMY.

- If you want the Reorg function to collect the prefix resolution and update records in a predefined format, include a PRPURWF1 file. In this case, the DFSURWF1 file has the following meanings:
  - If DFSURWF1 is DUMMY, the secondary indexes are initialized with zero records.
  - If DFSURWF1 is omitted or a physical data set, DFSURWF1 is ignored and prefix update records are written to PRPURWF1.
Unused files

Some files normally used in a traditional reorganization (DFSURGU1 and DFSUINPT) are not necessary with the Reorg function. If these files are specified in the JCL, the Reorg function will take special action as the following sections describe.

**DFSURGU1**

This sequential file is used during a traditional unload to contain the unloaded data set. With the Reorg function, the DFSURGU1 file is obsolete. If you need the sequential unload data set for other purposes, specify DFSURGU2. This file is in standard HD Unload format. See “DFSURGU2 DD” on page 360.

**DFSUINPT**

The DFSUINPT file is used during a traditional reload as input to the reload process. It is normally the file created by the unload process (DFSURGU1). If you specify INPUT(DB), this file is obsolete and the Reorg function will deallocate the DD statement if it is found in the JCL.

If you specify INPUT(UNLDFILE), the DFSUINPT DD statement or the appropriate number of DFSUINnn DD statements are required.

**Keywords**

The Reorg function uses the REORG command. The keywords that are available in the Reorg function are listed in Table 5.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLOWLC</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>407</td>
<td>Specify whether to allow logical children with no logical parents.</td>
</tr>
<tr>
<td>ALLOWLP</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>407</td>
<td>Specify whether to allow logical parents with no logical children.</td>
</tr>
<tr>
<td>AOI</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>409</td>
<td>Use the Online Reorg function automated operator interface during a Reorg function reorganization.</td>
</tr>
<tr>
<td>AUTHERR</td>
<td>(WTOR) (ABEND)</td>
<td>(ABEND)</td>
<td>410</td>
<td>Specifies the Reorg function action if DBRC authorization fails.</td>
</tr>
<tr>
<td>BUFSPACE</td>
<td>(1-999)</td>
<td>(20)</td>
<td>413</td>
<td>Specify maximum amount of buffer space (in MB) for the unload step.</td>
</tr>
<tr>
<td>Keyword</td>
<td>Accepted values</td>
<td>Default values</td>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------</td>
<td>----------------</td>
<td>------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CKUPDATE</td>
<td>(YES,STOPnnnn)</td>
<td>(YES,STOP4094)</td>
<td>414</td>
<td>Specify whether to check the RECON data sets for updates after the unload file was created.</td>
</tr>
<tr>
<td></td>
<td>(YES,OKAYnnnn)</td>
<td>(YES,OKAYnnnn)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREORG</td>
<td>(YES,OKAYnnnn)</td>
<td>(YES,OKAYnnnn)</td>
<td>419</td>
<td>Specify conditional reorganization option for MAXM Database Advisor for IMS.</td>
</tr>
<tr>
<td></td>
<td>(YES,STOPnnnn)</td>
<td>(YES,STOPnnnn)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD</td>
<td>(dbName1,</td>
<td>none</td>
<td>420</td>
<td>Specify the DBD name. Support for the DBD keyword was added for use with the GROUP subcommand.</td>
</tr>
<tr>
<td></td>
<td>dbName2,...)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBRC</td>
<td>(YES)</td>
<td>(YES,OKAYnnnn)</td>
<td>422</td>
<td>Specify whether to invoke DBRC. For HALDBs, DBRC(YES) is required.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DESCICODE</td>
<td>(nn,...)</td>
<td>(7)</td>
<td>490</td>
<td>Specify descriptor codes used to route messages to the console.</td>
</tr>
<tr>
<td></td>
<td>max 10 entries</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFSDBUX1</td>
<td>(YES,YES)</td>
<td>(YES,YES)</td>
<td>424</td>
<td>Specify whether to invoke the DFSDBUX1 user exit.</td>
</tr>
<tr>
<td></td>
<td>(NO,NO)</td>
<td>(NO,NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(YES,NO)</td>
<td>(YES,NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(NO,YES)</td>
<td>(NO,YES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>(YES)</td>
<td>(NO)</td>
<td>430</td>
<td>Dynamically allocate all database data sets using the data set names from the RECON data sets for the original database data sets and as a model for the shadow data sets. For HALDBs, DYNALLOC(YES) is required; it is also the default.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(YES,YES,x)</td>
<td>(YES,YES,x)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(YES,NO, x)</td>
<td>(YES,NO, x)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(NO,YES)</td>
<td>(NO,YES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EOJACCES</td>
<td>(EX)</td>
<td>(EX)</td>
<td>433</td>
<td>Access level for final /STA command.</td>
</tr>
<tr>
<td></td>
<td>(UP)</td>
<td>(UP)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(RD)</td>
<td>(RD)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(RO)</td>
<td>(RO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXPAND</td>
<td>(YES)</td>
<td>(NO)</td>
<td>415</td>
<td>Expand compressed segments.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FBFF</td>
<td>(n1,n2...n10)</td>
<td>(n1,n2...n10)</td>
<td>434</td>
<td>Specify an override value for the free block frequency factor specified in the DBD</td>
</tr>
<tr>
<td>FMTALL</td>
<td>(YES)</td>
<td>(NO)</td>
<td>438</td>
<td>Format all data sets.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FMTRAAG</td>
<td>(YES)</td>
<td>(YES)</td>
<td>439</td>
<td>Format the HDAM root addressable area.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FORCELP</td>
<td>(Y)</td>
<td>(N)</td>
<td>439</td>
<td>Specify whether to force logical parent counter fields.</td>
</tr>
<tr>
<td></td>
<td>(N)</td>
<td>(N)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRFPINT</td>
<td>(ddname)</td>
<td>BMCPINT</td>
<td>522</td>
<td>Specify the data set that will contain the Online Reorg function and Reorg function reports and processing output.</td>
</tr>
<tr>
<td>FSPF</td>
<td>(n1,n2...n10)</td>
<td>95</td>
<td>441</td>
<td>Specify a percentage of free space to qualify a block as free.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDAMFSPF</td>
<td>(IGNORE)</td>
<td>(NO)</td>
<td>443</td>
<td>Ignore the free space percentage factor.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 5  Reorg function keywords (part 3 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDSORT</td>
<td>(YES) (NO) (AUTO)</td>
<td>(AUTO)</td>
<td>443</td>
<td>Invoke the HD Sort utility or the BMC RAP Sequencer.</td>
</tr>
<tr>
<td>ICNEEDED</td>
<td>(ON) (OFF)</td>
<td>(ON)</td>
<td>457</td>
<td>Specify whether to turn off the ICNEEDED flag for all NONRECOV DBDs.</td>
</tr>
<tr>
<td>ICP</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>458</td>
<td>Invoke the Image Copy function. If you code ICP(Y), you must also provide</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>the ICPSYSIN data set. For more information, see Chapter 11, “Image Copy</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>function.”</td>
</tr>
<tr>
<td>IDCAMS</td>
<td>(input) (output) (input, output, Y)</td>
<td>none</td>
<td>464</td>
<td>Specify SYSIN and SYSPRINT files for IDCAMS processing.</td>
</tr>
<tr>
<td></td>
<td>(input) (output, N) (model)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ILDS</td>
<td>(AUTO) (BUILD) (INIT)</td>
<td>(AUTO)</td>
<td>468</td>
<td>Specify to build or initialize Indirect List Data Sets.</td>
</tr>
<tr>
<td>IMSID</td>
<td>(imsid)</td>
<td>Use the IMSID</td>
<td>470</td>
<td>Specify IMSID under which reports are stored in the PDX.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>from IMS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>generation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DFSVC000</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>module).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPUT</td>
<td>(DB) (UNLDFILE)</td>
<td>(DB)</td>
<td>473</td>
<td>Specify the input for the reorganization (database or HD Unload file).</td>
</tr>
<tr>
<td>LDPPRINT</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>522</td>
<td>Specify the data set that will contain the Reload function reports and</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>processing output.</td>
</tr>
<tr>
<td>LPCK</td>
<td>(NO) (YES)</td>
<td>(YES)</td>
<td>487</td>
<td>Retrieve the logical parent concatenated key (LPCK).</td>
</tr>
<tr>
<td>MONITOR</td>
<td>(nnnnnn)</td>
<td>no automatic</td>
<td>490</td>
<td>Monitor job progress.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>monitoring</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MONUSERS</td>
<td>(user01, user02, ...)</td>
<td>Send monitor</td>
<td>490</td>
<td>Send monitor messages to up to 10 specified TSO user IDs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>messages to TSO</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>users and</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BMCMSG.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOAUTH</td>
<td>(RUN) (STOP)</td>
<td>(STOP)</td>
<td>494</td>
<td>Specify the action to take if database authorization is prohibited.</td>
</tr>
<tr>
<td>OLDDBD</td>
<td>(ddname)</td>
<td>none</td>
<td>496</td>
<td>Change the DBD definition during a reorganization.</td>
</tr>
<tr>
<td>OSAMINIT</td>
<td>(YES) (NO)</td>
<td>(YES)</td>
<td>499</td>
<td>Specify the reset of the high-used RBA for a database that has multi-volume</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>OSAM data sets.</td>
</tr>
<tr>
<td>OSAMMAX</td>
<td>(4) (8)</td>
<td>(4)</td>
<td>500</td>
<td>Specify the size limit of an OSAM database.</td>
</tr>
</tbody>
</table>
### Table 5  Reorg function keywords (part 4 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OVFLONLY</td>
<td>(segname1, se name2,...) max 25 names</td>
<td>none</td>
<td>503</td>
<td>Place specified segments in HDAM overflow area.</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>(YES)(NO)</td>
<td>(YES)</td>
<td>504</td>
<td>Enable parallel processing.</td>
</tr>
<tr>
<td>PART</td>
<td>(ALL)</td>
<td>(ALL)</td>
<td>508</td>
<td>Perform reorganization tasks for one or more database partitions. The partition name is the ddname.</td>
</tr>
<tr>
<td>PDX</td>
<td>(OKAYnnnn)</td>
<td>(OKAY0004)</td>
<td>518</td>
<td>Specify action to take if the PDX fails to open. For HALDBs, PDX is not supported.</td>
</tr>
<tr>
<td>PDXPARMS</td>
<td>(userName)</td>
<td>Use the DBD name specified in the EXEC statement’s PARM operand.</td>
<td>520</td>
<td>Override the DBDname member in the PDX.</td>
</tr>
<tr>
<td>REPORTS</td>
<td>See page 526</td>
<td>See page 526</td>
<td>526</td>
<td>Generate reports.</td>
</tr>
<tr>
<td>PRTOLC</td>
<td>(Y)(N)</td>
<td>(Y)</td>
<td>521</td>
<td>Specify whether to print logical child records with no logical parent.</td>
</tr>
<tr>
<td>PRTOLP</td>
<td>(Y)(N)</td>
<td>(N)</td>
<td>521</td>
<td>Specify whether to print logical parents with no logical children.</td>
</tr>
<tr>
<td>RMBYTES</td>
<td>(SEGSIZE)</td>
<td>(MINBYTES)</td>
<td>529</td>
<td>Specify how to calculate the database record size to determine when to place consecutively inserted segments into overflow.</td>
</tr>
<tr>
<td>ROUTCODE</td>
<td>(nn,...) max 10 entries</td>
<td>(2,7)</td>
<td>530</td>
<td>Specify routing codes to control destination of messages sent to the console.</td>
</tr>
<tr>
<td>SCAN</td>
<td>(-nnn+mmmm) (+nnn-mmmm) (-+nnn),(+-mmmm)</td>
<td>(0) for HIDAM (-10+10) for HDAM</td>
<td>531 531</td>
<td>Set space search method.</td>
</tr>
<tr>
<td>SEGPOS</td>
<td>(segname1,n1, segname2,n2,...) max 25 names</td>
<td>none</td>
<td>535</td>
<td>Specify which segment type to place near the root.</td>
</tr>
<tr>
<td>SEGPOSO</td>
<td>(ALL)</td>
<td>none</td>
<td>536</td>
<td>Specify whether to place a segment in the HDAM overflow area, if it exceeds the limit specified in the SEGPOS specification.</td>
</tr>
<tr>
<td>SHARE</td>
<td>(YES)(NO)(YES,READ)</td>
<td>(NO)</td>
<td>540</td>
<td>Specify whether users will have access to the database during reorganization.</td>
</tr>
<tr>
<td>SIUPRINT</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>522</td>
<td>Specify the data set that will contain the Index Build function reports and processing output.</td>
</tr>
</tbody>
</table>
You can include your preferred defaults for most REORG keywords in a global options module. The Online Reorg function and the Reorg function share the global options module. To set the global options, use the JCL member #FRFGLBL in the sample library. Executing this job creates a global options module named FRF$GLBL.

No ISPF interface is available for the Reorg function.

Table 6 describes the keywords that you can use in the global options module.

### Table 6 Global options module keywords (part 1 of 2)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMSOUT</td>
<td>You can specify a default data set name for the AMSOUT data set. This keyword is not available as a PLUSIN keyword.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTHERR</td>
<td>If you want the Reorg function to issue a WTOR when DBRC authorization fails, specify AUTHERR(WTOR).</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 6  Global options module keywords (part 2 of 2)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXPAND</td>
<td>The Reorg function ignores the EXPAND keyword specification in the global options module. If you want to expand data during the unload process, specify EXPAND(YES) with the REORG command.</td>
</tr>
<tr>
<td>ICNEEDED</td>
<td>The NOTIFY.REORG command turns the ICNEEDED flag on. If you want the ICNEEDED flag left on for databases that are registered as NONRECOV, specify ICNEEDED(ON). If you do not want the ICNEEDED flag left on for these databases, specify ICNEEDED(OFF) or do not include the ICNEEDED keyword in the global options module.</td>
</tr>
<tr>
<td>NOAUTH</td>
<td>If you want the Reorg function execution to continue if the database was set to NOAUTH, specify NOAUTH(RUN). If you want the Reorg function execution to stop, specify NOAUTH(STOP).</td>
</tr>
<tr>
<td>SHARE</td>
<td>If you want to allow other users read access to the database during reorganization, specify SHARE(YES). If you do not want to allow other users read access, specify SHARE(NO) or do not include the SHARE keyword in the global options module.</td>
</tr>
</tbody>
</table>
| SWAP    | If you want automatic data set name swapping as the default, specify SWAP(YES). If you want to swap the data set names manually, specify SWAP(NO). If you want the Reorg function to swap the data set names and delete the old data sets, specify SWAP(D). Carefully consider using SWAP(D) as a global option; you could inadvertently delete data sets that you might need.  

Note: No global option for SWAP(x,IC) exists. You must specify this keyword in PLUSIN. |
| UNLDD   | You can specify default substitution characters for the input ddnames. |

Table 7  Reorg function reports

<table>
<thead>
<tr>
<th>Report</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD Change Summary</td>
<td>592</td>
</tr>
<tr>
<td>Dataset List</td>
<td>591</td>
</tr>
<tr>
<td>IDCAMS System Services</td>
<td>600</td>
</tr>
<tr>
<td>Message Log</td>
<td>604</td>
</tr>
<tr>
<td>Options in Effect</td>
<td>604</td>
</tr>
</tbody>
</table>

Reports

The Reorg function generates the reports shown in Table 7. The functions performing the reorganization also generate the appropriate reports.
Post-processing phase

Post-processing is the time during the reorganization after the database has been reorganized but before the database is ready for access by users. Because the database has been reorganized, the post-processing tasks must be completed. During post-processing, messages are issued that report the beginning and completion of each task. These messages are written to the BMCMSG data set.

You can delay post-processing tasks; if you do, you must execute the restart process to complete them. You must use the restart process if any post-processing tasks fail. For more information, see Appendix B, “Post-processing.”

**NOTE**

If you specify INPUT(UNLDFILE), you cannot delay post-processing tasks. However, if a failure occurs during DBRC processing, you can restart post-processing tasks.

Post-processing includes the following tasks:

- data set name swapping
- DBRC notifications
- restart processing, if needed

Empty database initialization

Sometimes you need an empty database, particularly in an online environment. Because IMS does not allow the creation of a database online, the database must be created offline. Creating an empty database offline usually involves inserting one root segment and deleting it. The Reorg function allows you to initialize the database so it can be used online.

You can execute the REORG command to perform this initialization function with a JCL statement modification. To create a usable empty database, specify a DFSUINPT DD statement that points to an empty sequential data set. You cannot specify DFSUINPT DD DUMMY. Include the INPUT(UNLDFILE) keyword in the PLUSIN control statement data set. For more information, see “INPUT” on page 473.
Operational considerations

This section describes how to set options to deal with authorization failures. It also explains how to allow read access to the database during the reorganization and how to set options to allow the Reorg function to swap data set names. The section also provides information concerning secondary indexes and image copies.

DBRC authorization

Before any DBRC authorization, the Reorg function checks all data set names for logical errors. Three sources of data set names are available: the data set names registered to DBRC, the input data set names in the JCL, and the output data set names in the JCL. Table 8 shows the possible combinations.

Table 8  Valid data set name combinations for authorization

<table>
<thead>
<tr>
<th>Data set names and DBRC registration</th>
<th>Reorg function keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td>■ No output data set names are registered to DBRC.</td>
<td>SHARE(YES)</td>
</tr>
<tr>
<td>■ All input data set names are registered to DBRC.</td>
<td></td>
</tr>
<tr>
<td>■ No output primary database data set names are registered to DBRC.</td>
<td>SHARE(NO)</td>
</tr>
<tr>
<td>■ All output secondary index data set names are registered to DBRC</td>
<td></td>
</tr>
<tr>
<td>■ All output secondary index data set names are specified.</td>
<td></td>
</tr>
<tr>
<td>■ All input primary database data set names are registered to DBRC.</td>
<td></td>
</tr>
<tr>
<td>■ All output secondary index data set names are registered to DBRC</td>
<td>SHARE(NO) and</td>
</tr>
<tr>
<td>■ All output secondary index data set names are specified.</td>
<td>SWAP(x,IC)</td>
</tr>
<tr>
<td>■ No output primary database data set names are specified.</td>
<td></td>
</tr>
<tr>
<td>All output data set names are registered to DBRC.</td>
<td>SHARE(NO) and</td>
</tr>
<tr>
<td>■ All output secondary index data set names are registered to DBRC</td>
<td>INPUT(UNLDFILE)</td>
</tr>
<tr>
<td>■ All output secondary index data set names are specified.</td>
<td></td>
</tr>
<tr>
<td>■ No output primary database data set names are specified.</td>
<td></td>
</tr>
</tbody>
</table>

The Reorg function also checks the status of the database. If the status is BACKOUT NEEDED or RECOV NEEDED, reorganization is not allowed.

If an error occurs during the data set name swapping process, the Reorg function sets the DBRC NOAUTH flag.
**DBRC authorization failures**

The AUTHERR keyword specifies the action that the Reorg function takes if DBRC authorization fails. If you specify AUTHERR(ABEND) and DBRC authorization fails, the job terminates with user abend code U0047. No updates are made to the database. When you resolve the reason for the authorization failure, you can resubmit this job by using the REORG command.

If you specify AUTHERR(WTOR) and DBRC authorization fails, the Reorg function sends a message to the operator, requesting some action. For more information, see “AUTHERR” on page 410. This feature is useful in job networks, which would have to be rescheduled if the operator could not manipulate the current jobs.

The AUTHERR keyword applies only when authorization is checked during initialization processing. The Reorg function always requires exclusive control during the data set name swapping process.

**Database prohibits authorization**

The NOAUTH keyword specifies the action that the Reorg function takes if the database status in DBRC prohibits authorization. This situation occurs if you have issued a DBRC command like CHANGE.DB DBD(dbdName) NOAUTH.

If you intend to block DBRC authorization for everything except the reorganization by issuing the CHANGE.DB DBD(dbdName) NOAUTH command, use NOAUTH(RUN). The authorization process for the Reorg function ignores the DBRC NOAUTH flag that you set.

If an error occurs during data set name swapping or if the DBD definition changes during the reorganization (OLDDBD keyword) and you specified SWAP(YES) or SWAP(D), the Reorg function sets the DBRC NOAUTH flag.

**Secondary indexes**

If the database has secondary indexes, you have the following choices in how the Reorg function handles them:

- Use the same (old) data set names for output.

  You must ensure the secondary index data sets are empty (by using DELETE/DEFINE in a previous step) before the Reorg function tries to use them. You must specify SHARE(NO).
Preserve the original index data sets and use temporary data sets for output.

To preserve the original index data sets that are associated with the disorganized database, use new data set names. The Reorg function recognizes that the index data sets with new names are reorganized data sets, and the Reorg function includes them in the swap process. The primary reason you would do this is to ensure that you can easily fall back to the prereorganization status of the database if you encounter problems during or after the reorganization process, or if you specify SHARE(YES).

If any output data sets are not empty, the Reorg function issues an error message and the job step terminates.

If a reorganization fails and you specified SHARE(NO) and used old data set names for the secondary index data sets, your original indexes will be corrupted. The secondary index files will be empty or will not match the database before the start of the reorganization process.

To get indexes identical to the ones you had before beginning the reorganization, you must perform one of the following tasks:

- Recover the secondary index files.
- Use the Index Build function to scan the disorganized primary database and build the secondary index files.

**NOTE**

If you specified INPUT(UNLDFILE), the disorganized primary database is no longer intact. Use the image copy that you took before the reorganization.

You can prevent the loss of the secondary indexes of the disorganized database by taking either of the following actions:

- Specify different data set names in the JCL for the reorganized secondary index data sets.
- Rename the original secondary index files, or make a backup of the original secondary index files.

**Image copies**

Some image copies that you take in a traditional reorganization might not be necessary when you use the Reorg function. This section lists guidelines for taking image copies with a Reorg function reorganization.
After the reorganization, image copies are required because they establish post-reorganization recovery points. If necessary, you can leave the database in read mode and take the image copy while allowing users read-only access.

If you specify ICP(YES), the Reorg function calls the Image Copy function to take the post-reorganization image copy automatically as the database is written.

For more information, see Chapter 11, “Image Copy function.”

**Recovery if the reorganization fails**

Some sites take an image copy of a disorganized database before reorganizing that database. Usually, these image copies are taken to simplify recovery if the reorganization fails. With the Reorg function, this image copy might not be necessary because the disorganized database is fully available and intact until the data set name swapping process.

---

**WARNING**

If you specify INPUT(UNLDFILE), BMC recommends that you take an image copy before you begin the reorganization. Because INPUT(UNLDFILE) reorganizes the database to the original data sets, the original database will become invalid for recovery purposes if a failure occurs during the reorganization.

---

The data set name swapping process activates the reorganized (shadow) database data set names, making the disorganized data sets obsolete. The disorganized data sets should not be used any more. You might want to use them as a backup for other reasons; however, if you do, you also need to issue DBRC commands such as NOTIFY.RECOV.

When you specify SWAP(YES,IC), the reorganized image copies of the primary database and primary index are restored to the original (disorganized) data set.

When you specify SWAP(x,IC) or use the secondary index data set names that are registered to DBRC, you rebuild secondary indexes to the original data set names. No shadow data sets of the rebuilt secondary indexes are created. You must recover the secondary indexes if the reorganization fails:

- If the reorganization fails before post-processing tasks begin, you must recover the secondary indexes in either of the following ways:

  — Execute the Index Build function.
  — You can take a pre-reorganization image copy of the secondary indexes to simplify the recovery process. Apply the image copy (and logs, if any).

- If the reorganization fails after post-processing tasks begins (message BMC90043I notifies you of post-processing status), you must execute the RESTART command (or Restart Swap utility) to complete the post-processing tasks.
Database replication

In most cases, the Reorg function will not execute if the original data sets and the shadow data sets are in different MVS catalogs. However, if you want to replicate the database, and leave the original database disorganized, you can have the original and reorganized data sets in different MVS catalogs. To replicate the database, complete the following steps:

1. Specify SWAP(NO).

2. Specify AMSOUT DD DUMMY, or ensure that an AMSOUT file is not allocated for the job step. An AMSOUT file can be allocated by the FRF$GLBL global options module or a DD statement.
Online Reorg function

This chapter describes how to use MAXM Reorg/Online to reorganize a database while allowing updates. This chapter includes the following topics:

Overview ................................................................. 109
Requirements ......................................................... 113
Handling BMPs with AR/CTL ........................................ 114
JCL requirements ...................................................... 115
DD statements ......................................................... 116
Keywords ............................................................. 119
PLUSIN examples .................................................... 123
Global options module .............................................. 124
Report ................................................................. 124
Unload phase ......................................................... 125
Post-processing phase .............................................. 125
Automated operator interface .................................... 126
BMPs ................................................................. 127
DBR commands ....................................................... 128
DBD changes during reorganization .............................. 129
  DELTA IMS DB/DC installed .................................... 130
  DELTA IMS DB/DC not installed .............................. 132
Database recovery ................................................ 133
Non-disruptive maintenance .................................... 133

Overview

To enable an online reorganization, use the following functions:

- Online Reorg function
- Reorg function
- Prefix resolution and update function in the Reorg function
The Online Reorg function requires full DBRC control. DBRC must be active, and all databases must be registered. BMC recommends that you register all databases and database data sets and that you turn the FORCER flag on. You can accomplish these tasks by using the IMS DBRC utility (DSPURX00).

To invoke the Online Reorg function, use the SHARE(YES,UPDATE) keyword with the REORG command. All reorganization activities are executed and completed in the same job step.

**NOTE**

If you are reorganizing multiple databases at the same time, you must also specify the GROUP subcommand. For more information, see Chapter 5, “Multiple databases.”

When you invoke the Online Reorg function, the following events take place:

1. The Online Reorg function initializes.

   During initialization, the Online Reorg function initiates communication with the Reorg function.

2. The Reorg function checks the JCL for logic errors.

3. The original database is read, using one of two methods (based on your specifications):
   - Using the Copy function

      The Online Reorg function issues a /BMCPAU and /STA command sequence to create a sync point and to start capturing log records. The copy engine copies the original database to the shadow database. When the copy is complete, the Online Reorg function issues another /BMCPAU and /STA command sequence. The captured log records are applied to the shadow databases. At this point, log record capturing is turned off and database change capturing is started.

   - Using the Snapshot feature

      Specify SNAPSHOT(Y). The Online Reorg function issues a /DBR and /STA command sequence to create a sync point and to start capturing database changes.
The Snapshot feature facilitates the unload by reading the original database without having to create a copy.

**NOTE**

BMC recommends this method. It requires less DASD and eliminates one of the /DBR and /STA pairs. This method requires the SNAPSHOT UPGRADE FEATURE or XBM product. (For details about installing the Snapshot feature, see the [BMC Products for IMS Installation Guide](#) and the [EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE Installation Guide](#).)

**NOTE**

When you specify INPUT(DB,INIT), the reorganized database is initialized as an empty database. Therefore, the original database is not read or copied. There is no capture of log records or database changes.

4. The Reorg function receives control and begins the reorganization by invoking the following functions:

- Unload function
- Reload function
- Limited prefix resolution and update
- Index Build function
- Image Copy function

The reorganization depends on the method used to read the original database. If SNAPSHOT is used, the Unload function reads the original database and passes the segments to the Reload function. No intermediate unload file is required. The Unload function, Reload function, Index Build function, Prefix Resolution/Update function, and Image Copy function overlap.

If SNAPSHOT is not used, the shadow database, which was created in the copy phase, is unloaded to a temporary unload file. When the unload is complete, the Reload function reads the unload file and reloads the shadow database. The Index Build function, Prefix Resolution/Update function, and Image Copy function overlap with the Reload function.

When the database is being initialized, the Unload function passes an empty database indication to the Reload function. The Reload function initializes the shadow database. The Index Build function overlaps the Reload function and initializes the secondary indexes. The Image Copy function overlaps the Reload function.

5. After the database has been reorganized, the Online Reorg function uses an IMS batch program that applies the updates that were made to the original database to the reorganized shadow database. This is done in several iterations until the number of updates left to be applied is fairly small.
6. After most updates have been applied to the reorganized shadow database, the Online Reorg function performs the following tasks:

   — issues a /DBR
   — applies the remaining updates (except when the database is being initialized)
   — swaps data set names
   — issues DBRC commands
   — stops recording (except when the database is being initialized)

   While these tasks take place, no one is authorized to access the database. These tasks should take a minute or less, under normal circumstances.

   If you do not want a log switch to occur, specify DBR(NOFEOV). For more information, see “DBR” on page 421.

7. After these tasks have completed successfully, the Online Reorg function makes the database available by issuing a /STA command. The batch logs that were created by the IMS batch program are processed to change the time stamp. When the database is being initialized, there are no batch logs to process.

**Figure 12** shows the system flow for MAXM Reorg/Online.

**Figure 12  MAXM Reorg/Online system flow**
Requirements

Depending on your organization’s needs, reorganizing a database requires the following actions:

- **Required.** Install MAXM Reorg/Online. If you want the Image Copy function to invoke POINTER CHECKER PLUS® to check pointers during the image copy, install the version of POINTER CHECKER PLUS with the same maintenance date. For instructions, see the *BMC Products for IMS Installation Guide*.

- **Optional.** If you use the Online Reorg function in a CICS/DBCTL environment, you must install certain modules. For more information, see the *BMC Products for IMS Installation Guide*.

- **Required.** Create and submit one set of JCL that runs the entire reorganization process.

---

**NOTE**

To use existing FRF or MAXM Reorg classic JCL, you must modify the JCL as follows:

- For dynamic allocation (DYNALLOC), remove the original database data sets from the JCL.

- Add SHARE(YES,UPDATE) and the following keywords to the PLUSIN control statement data set:
  
  — Specify the IDCAMS keyword or pre-allocate the shadow data sets. For more information, see “IDCAMS” on page 464.

  — Specify DYNALLOC(YES,YES,x) if VSAM data sets do not exist.

---

- **Optional.** If you want any special processing, set keywords in the PLUSIN DD statement. For an overview of the applicable keywords, see Table 96 on page 390.

---

**NOTE**

BMC supports reorganizing (and copying) multiple databases simultaneously. For detailed information about this feature, see Chapter 5, “Multiple databases.”
Handling BMPs with AR/CTL

For automatic handling of batch message processing (BMP) programs during the Online Reorg/Concurrent Reorg process, the CRG Copy utility can use the suspend-and-resume interface of the APPLICATION RESTART CONTROL (AR/CTL™) for IMS product from BMC. The CONCURRENT REORG for IMS product performs the communication between AR/CTL and the Image Copy utility.

To use this optional suspend-and-resume interface, perform the following steps:

1. Install AR/CTL as explained in the APPLICATION RESTART CONTROL Installation Guide.

2. Ensure that the MXO or CRF authorization password is available to the BMC Consolidated Subsystem (BCSS) that supports the Application Enhancement Series (AES) products and AR/CTL. You can include the library that contains the password module in the AESPAUTH DD statement concatenation in the BCSS startup procedure, or you can copy the password module to a library that is already in the AESPAUTH DD concatenation. For more information, see the AR/CTL documentation set.

3. Verify that AR/CTL and CONCURRENT REORG are implemented correctly by checking for the following messages:

   The following IMSCTL region message:

   BMC90489I CHANGE RECORDING FACILITY INITIALIZATION COMPLETED

   The following AES/BMCBCSS message:

   BMC151071I THE AR/CTL FOR IMS INTERFACE WITH IMAGE COPY PLUS IS ENABLED.

   The following Image Copy job messages:

   BMC90486I ARCTR STOP REQUEST ....
   BMC90486I ARCTR RESTART REQUEST ....

   The following BMP job (under control of AES/ARC) messages:

   BMC74448I AES/ICP INTERFACE SUSPENSION ...
   BMC74448I AES/ICP INTERFACE RESUMPTION ...
   BMC74413I REATTACHING PROCESS ...
4 Code the Concurrent Reorg utility job step.

Ensure that the AES execution load module library is available to the CRG step.
You can add these libraries to the STEPLIB, JOBLIB, or LNKLIST concatenation.

5 In each BMP that you want AR/CTL to control, implement AR/CTL
checkpoint/restart services as described in the APPLICATION RESTART
CONTROL User Guide.

6 Execute the Concurrent Reorg job step.

The job step must be APF-authorized. You can APF-authorize the load module
library or use the DBUSS.

JCL requirements

This section describes the JCL requirements for the Online Reorg function.

NOTE
When you specify the SHARE=YES,UPDATE) keyword (to invoke the Online Reorg function),
do not specify a DD statement for the original database.

This JCL shows two steps: the first step allocates an OSAM shadow output data set,
and the second step reorganizes the database. The PLUSIN control statement shows
the REORG command, the SHARE=YES,UPDATE) keyword, and optional keywords.

Figure 13  Online Reorg function JCL (part 1 of 2)

```plaintext
//***************************************************************************
//** ONLINE REORG SAMPLE - REORG JCL USING MXO
//** HIDAM/OSAM DATABASE
***************************************************************************
/**
**/MXO      EXEC PGM=DFSRRC00,REGION=0M,
   //PARM='ULU,DFSURGLO,PRIMEDB,,,,,,,,,,,Y'
//STEPLIB  DD  DISP=SHR,DSN= bmc.xxx.load
// DFSRESLB DD  DISP=SHR,DSN=IMSVS.RESLIB
// DFSURCDS DD  DISP=SHR,
// DSN=FROM.DATABASE.PREREORG.STEP
// IMS      DD  DISP=SHR,DSN=IMSVS,DBDLIB
// AMSOUT   DD  DISP=SHR,DSN=your.RESTART.INFO.DATASET
// CRFRDER  DD  DISP=(,CATLG,CATLG),DSN=your.CRFRDER,
// UNIT=SYSDA,SPACE=(CYL,(3,1),RLSE)
// CRFRDER2 DD  DISP=(,CATLG,CATLG),DSN=your.CRFRDER2,
// UNIT=SYSDA,SPACE=(CYL,(3,1),RLSE)
// IEFRDER  DD  DISP=(,CATLG,CATLG),DSN=your.IEFRDER1
```
**DD statements**

The following DD statements are specific to the Online Reorg function REORG command.

**STEPLIB DD**

Required. If you want to use the interface to AR/CTL, include the AES execution load module library. For more information, see “Automated operator interface” on page 126. All libraries in the STEPLIB must be APF-authorized.

---

**Figure 13** Online Reorg function JCL (part 2 of 2)

```
//    UNIT=SYSDA,SPACE=(CYL,(3,1),RLSE)
//IEFRDER2 DD DISP=(,CATLG,CATLG),DSN=your.IEFRDER2,
// UNIT=SYSDA,SPACE=(CYL,(3,1),RLSE)
//IDCIN   DD DISP=SHR,DSN=YOUR.IDCAMS.DELETE.DEFINES
//IDCOUT  DD SYSOUT=* 
//*/
///* DO NOT SPECIFY THESE SHADOW DD STATEMENTS WHEN DYNALLOC(Y,Y,Z)
//*/PRIMEDB DD DISP=SHR,DSN=MXO.OSAM.DATABASE.DATASET.Z
//*/PRIMIDX DD DISP=SHR,DSN=MXO.VSAM.DATABASE.DATASET.Z
/*/ /* NEVER SPECIFY THE FOLLOWING ORIGINAL DD STATEMENTS WHEN RUNNING MXO/CRF
//*/OLDMEB DD DISP=SHR,DSN=MXO.OSAM.DATABASE.DATASET
//*/OLDMIDX DD DISP=SHR,DSN=MXO.VSAM.DATABASE.DATASET
/*/ /*PLUSIN DD *
REORG SHARE(Y,UPDATE) IDCAMS(IDCIN,IDCOUT) DYNALLOC(Y,Y,Z)
/*
/** IC/AIC CONTROL STATEMENTS ARE REQUIRED FOR ALL RECOV DATABASES
/** VIC(Y) IS ALSO SUPPORTED
/**ICPSYSIN DD *
IC DBD(PRIMEDB) DDN(PRIMEDB) IC(IC1)
IC DBD(PRIMIDX) DDN(PRIMIDX) IC(IC2)
/*
/** IMAGE COPY DD STATEMENTS ARE REQUIRED FOR ALL RECOV DATABASES
/** UNLESS YOU DYNAMICALLY ALLOCATE THE OUTPUT IMAGE COPY DATA SETS
/** YOU CANNOT STACK IMAGE COPIES; IF NOT ENOUGH TAPE DRIVES; USE DASD
//*/IC1      DD DISP=(,CATLG,DELETE),
// DSN=YOUR.PRIMEDB.IC,
// UNIT=SYSALLDA,SPACE=(CYL,(100,10),RLSE)
//*/
//*/IC2      DD DISP=(,CATLG,DELETE),
// DSN=YOUR.PRIMIDX.IC,
// UNIT=SYSALLDA,SPACE=(CYL,(100,10),RLSE)
//*/
//BMCTRACE DD SYSOUT=* 
//BMCM5G DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 
//*/
```
IEFRDER DD

Optional. If you specify the LOGDSN keyword, the Online Reorg function dynamically allocates this data set. This DD statement describes the log data set that is created during the change apply process. If you define a new data set, you must specify DISP=(NEW,CATLG,CATLG).

If you do not supply DCB information, the Online Reorg function uses 30 buffers and a block size equal to 32 KB (for tape) or half track (for disk). This DD is not used with INPUT(DB,INIT).

CRFRDER DD

Optional. If you specify the LOGDSN keyword, the Online Reorg function dynamically allocates this data set. This DD statement describes the log data set that is created during the log recover process. The CRFRDER data set contains modified information from the IEFRDER data set. The Online Reorg function creates the CRFRDER data set to provide the log records required for a recovery (if a recovery is needed). The CRFRDER data set is automatically registered to the RECON data set as a batch log. If you define a new data set, you must specify DISP=(NEW,CATLG,CATLG). This DD is not used with INPUT(DB,INIT).

IEFRDER2 DD

Optional. If you specify LOGDSN(dsnMask,DUAL), the Online Reorg function dynamically allocates this data set. If your installation requires dual logging, specify IEFRDER and IEFRDER2. If you define a new data set, you must specify DISP=(NEW,CATLG,CATLG).

If you do not supply DCB information, the Online Reorg function uses 30 buffers and a block size equal to 32 KB (for tape) or half track (for disk). This DD is not used with INPUT(DB,INIT).

CRFRDER2 DD

Optional. If you specify LOGDSN(dsnMask,DUAL), the Online Reorg function dynamically allocates this data set. If your installation requires dual logging, specify CRFRDER and CRFRDER2. If you define a new data set, you must specify DISP=(NEW,CATLG,CATLG). This DD is not used with INPUT(DB,INIT).

dbdddnam DD

Required if you do not use DYNALLOC(YES,YES,x). This DD statement describes the data set that will contain the database after it is reorganized (the shadow database). Because the Online Reorg function checks the catalog status of the data set at the beginning of the job step, you must define and catalog any OSAM data set before you run the Online Reorg function. VSAM data sets do not need to be preallocated if the IDCAMS keyword is specified.
Note the following considerations:

- If provided, the ddnames must match the ddnames in the DBD.
- The disorganized (original) database and the reorganized (shadow) database must be in the same catalog.
- For information about using multivolume data sets, see “Multivolume data sets” on page 225.
- If your SMS automatic class selection (ACS) routines check data set names, you must select the appropriate SMS or non-SMS volume for the shadow data sets. The original data sets and the shadow data sets must both be on an SMS or non-SMS volume. If you have the original data sets on an SMS volume and the shadow data sets on a non-SMS volume (or vice versa), data set name swapping fails.

**NOTE**

When using the Online Reorg function, do not include the database DDs in any step after the reorganization step. If you do, the online region might get a dynamic allocation error when the Online Reorg function issues the /STA command after the swap process.

**DFSVSAMP DD**

Optional. If the DD statement is not present, the Online Reorg function dynamically allocates the data set. Because the change apply program acts as an IMS batch program, a DFSVSAMP DD statement is required. The Online Reorg function dynamically allocates a DFSVSAMP data set with sufficient OSAM buffers, VSAM buffers, or both. If you include a DFSVSAMP DD statement in the JCL, the Online Reorg function honors all of the statements except those that specify a buffer pool.

**ICPSYSIN DD**

Optional. This DD contains the commands and keywords for the Image Copy function. The IC or AIC command is required for each database that is being reorganized and is registered as RECOV in DBRC. If the ICPSYSIN DD statement is omitted, the Online Reorg function will automatically create an image copy for all databases being reorganized that are registered as RECOV in DBRC. For more information, see “JCL requirements” on page 275.

**imagecpy DD**

Optional. You can specify Image Copy function keywords to dynamically allocate this data set. This DD statement describes the data set containing the image copy. For more information, see “JCL requirements” on page 275.
DFSURGU2 DD

Optional. If you include a DFSURGU2 DD statement that does not specify DUMMY in the JCL, the Online Reorg function writes the HD Unload file to the DFSURGU2 data set rather than to temporary data sets. If the DFSURGU2 data set is on DASD, you must provide a sufficient disk space specification in the JCL.

DFSURCDS DD

Required if there are logical relationships. This DD statement describes the control data set containing data generated by the IMS Prereorganization utility. If the database for reorganization does not contain logical relationships, this DD statement may be omitted.

iddeldef

Optional. This DD contains the IDCAMS statements for VSAM and OSAM shadow data sets. The ddname must match the ddname specified on the IDCAMS keyword.

Keywords

The Online Reorg function uses the REORG command with the SHARE(YES,UPDATE) keyword. Table 9 lists the keywords that are available for the Online Reorg function.

Table 9    Online Reorg function keywords (part 1 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLOWLC</td>
<td>(Y), (N)</td>
<td>N</td>
<td>407</td>
<td>Specify whether to allow logical children with no logical parents.</td>
</tr>
<tr>
<td>ALLOWLP</td>
<td>(Y), (N)</td>
<td>N</td>
<td>407</td>
<td>Specify whether to allow logical parents with no logical children.</td>
</tr>
<tr>
<td>AGN</td>
<td>(agnname)</td>
<td>none</td>
<td>407</td>
<td>Specify an application group name for the BMP.</td>
</tr>
<tr>
<td>AOIUSER</td>
<td>(progname)</td>
<td>none</td>
<td>410</td>
<td>Invoke a user exit from within the AOI.</td>
</tr>
<tr>
<td>CREORG</td>
<td>(YES,OKAYnnnn)</td>
<td>(YES,OKAYnnnn)</td>
<td>419</td>
<td>Specify conditional reorganization for MAXM Database Advisor for IMS.</td>
</tr>
<tr>
<td>DBD</td>
<td>(dbname1, dbname2,...)</td>
<td>none</td>
<td>420</td>
<td>Specify the DBD name. Support for the DBD keyword was added for use with the GROUP subcommand.</td>
</tr>
<tr>
<td>DBR</td>
<td>(FEOV), (NOFEOV)</td>
<td>(FEOV)</td>
<td>421</td>
<td>Specify whether to switch the OLDS for the /DBR at the end of the reorganization.</td>
</tr>
</tbody>
</table>
Table 9  Online Reorg function keywords (part 2 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBRC</td>
<td>(YES) (NO)</td>
<td>Use the DBRC specification in the IMSCTRL macro.</td>
<td>422</td>
<td>Specify whether to invoke DBRC. For HALDBs, DBRC(YES) is required.</td>
</tr>
<tr>
<td>DESC CODE</td>
<td>(mn,...)</td>
<td>(7)</td>
<td>490</td>
<td>Specify descriptor codes used to route messages to the console.</td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>430</td>
<td>Dynamically allocate all database data sets using the data set names specified in the RECON data sets, and use the DBRC-registered names as models for the shadows.</td>
</tr>
<tr>
<td>FBFF</td>
<td>(n1,n2...n10)</td>
<td></td>
<td>434</td>
<td>Specify an override value for the free block frequency factor specified in the DBD.</td>
</tr>
<tr>
<td>FMTALL</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>438</td>
<td>Format all data sets.</td>
</tr>
<tr>
<td>FMTRAA</td>
<td>(YES) (NO)</td>
<td>(YES)</td>
<td>439</td>
<td>Format the HDAM root addressable area.</td>
</tr>
<tr>
<td>FORCELP</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>439</td>
<td>Specify whether to force logical parent counter fields.</td>
</tr>
<tr>
<td>FRFPRINT</td>
<td>(ddname)</td>
<td>BMCPRINT</td>
<td>522</td>
<td>Specify the data set that will contain the Online Reorg function and Reorg function reports and processing output.</td>
</tr>
<tr>
<td>FSPF</td>
<td>(n1,n2...n10)</td>
<td>95</td>
<td>441</td>
<td>Specify a percentage of free space to qualify a block as free.</td>
</tr>
<tr>
<td>HDAMFSPF</td>
<td>(IGNORE) (NO)</td>
<td>(NO)</td>
<td>443</td>
<td>Ignore the free space percentage factor.</td>
</tr>
<tr>
<td>HDSORT</td>
<td>(YES) (NO)</td>
<td>(AUTO)</td>
<td>443</td>
<td>Invoke the HD Sort utility or the BMC RAP Sequencer.</td>
</tr>
<tr>
<td>ICNEEDED</td>
<td>(ON) (OFF)</td>
<td>(ON) for RECOV databases (OFF) for NONRECOV databases</td>
<td>457</td>
<td>Specify whether to turn off the ICNEEDED flag for all NONRECOV DBDs.</td>
</tr>
<tr>
<td>ICP</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>458</td>
<td>Invoke the Parallel Image Copy function.</td>
</tr>
<tr>
<td>IDCAMS</td>
<td>(input) (input,output) (input,output,Y) (input,output,N)</td>
<td>none</td>
<td>464</td>
<td>Specify SYSIN and SYSPRINT files for IDCAMS processing.</td>
</tr>
<tr>
<td>IMSID</td>
<td>(imsid)</td>
<td>Use the IMSID from IMS generation (DFSVC000 module).</td>
<td>470</td>
<td>Specify the IMSID under which reports are stored in the PDX.</td>
</tr>
<tr>
<td>Keyword</td>
<td>Accepted values</td>
<td>Default values</td>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------</td>
<td>--------------------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>INPUT</td>
<td>(DB,REORG)  (DB,INIT)</td>
<td>(DB,REORG)</td>
<td>473</td>
<td>Specify whether to reorganize the original database or initialize an empty database.</td>
</tr>
<tr>
<td>IRLMGRP</td>
<td>(irlmgrp)</td>
<td>none</td>
<td>477</td>
<td>Specify the IRLM group to be used with the Snapshot Copy function.</td>
</tr>
<tr>
<td>LDPPRINT</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>522</td>
<td>Specify the data set that will contain the Reload function reports and processing output.</td>
</tr>
<tr>
<td>LOGDSN</td>
<td>(dsname,SINGLE) (dsname,DUAL)</td>
<td>none</td>
<td>483</td>
<td>Dynamically allocate the IEFRDER and CRFRDER logs.</td>
</tr>
<tr>
<td>LOGEXPDT</td>
<td>(yyddd)</td>
<td>(yyyy/ddd)</td>
<td>485</td>
<td>Keep the logs until the specified date.</td>
</tr>
<tr>
<td>LOGRETPD</td>
<td>(mmmmmm)</td>
<td>none</td>
<td>485</td>
<td>Keep the logs the specified number of days.</td>
</tr>
<tr>
<td>LOGSMS</td>
<td>(YES,mgmt,strg, data) (NO)</td>
<td>Use the installation defaults.</td>
<td>485</td>
<td>Use SMS and the specified management, storage, and data classes.</td>
</tr>
<tr>
<td>LOGSPACE</td>
<td>(prim,sec)</td>
<td>(100,50)</td>
<td>486</td>
<td>Specify the amount of space to allocate when dynamically allocating the logs.</td>
</tr>
<tr>
<td>LOGUNIT</td>
<td>(unitName, unitName)</td>
<td>none</td>
<td>486</td>
<td>Use the specified unit names for the logs.</td>
</tr>
<tr>
<td>MONITOR</td>
<td>(nnnnnnnn)</td>
<td>no automatic monitoring</td>
<td>490</td>
<td>Monitor job progress.</td>
</tr>
<tr>
<td>MONUSERS</td>
<td>(user01, user02,...)</td>
<td>Send monitor messages to TSO users and BMCMSG.</td>
<td>490</td>
<td>Send monitor messages to up to 10 specified TSO user IDs.</td>
</tr>
<tr>
<td>OLDDBD</td>
<td>(ddname)</td>
<td>none</td>
<td>496</td>
<td>If you want to change the DBD definition during a reorganization, specify the OLDDBD definition here.</td>
</tr>
<tr>
<td>OSAMMAX</td>
<td>(4) (8)</td>
<td>(4)</td>
<td>500</td>
<td>Specify the size limit of an OSAM database.</td>
</tr>
<tr>
<td>OVFLONLY</td>
<td>(segname1, segname2,...)</td>
<td>none</td>
<td>503</td>
<td>Place specified segments in HDAM overflow area.</td>
</tr>
<tr>
<td>PART</td>
<td>(ALL) (partitionName, ...) (nn,nn,...) (nn)</td>
<td>(ALL)</td>
<td>508</td>
<td>Reorganize selected HALDB partitions.</td>
</tr>
<tr>
<td>PDX</td>
<td>(OKAYnnnnn) (STOPnnnn)</td>
<td>(OKAY0004)</td>
<td>518</td>
<td>Specify which action to take if the PDX fails to open.</td>
</tr>
<tr>
<td>PDXPARGS</td>
<td>(userName)</td>
<td>Use the DBD name specified in the EXEC statement’s PARM operand.</td>
<td>520</td>
<td>Override the DBD name member in the PDX.</td>
</tr>
</tbody>
</table>
Table 9  Online Reorg function keywords (part 4 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRTOLC</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>521</td>
<td>Specify whether to print logical child records with no logical parent.</td>
</tr>
<tr>
<td>PRTOLP</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>521</td>
<td>Specify whether to print logical parents with no logical children.</td>
</tr>
<tr>
<td>PSB</td>
<td>(psbName)</td>
<td></td>
<td>523</td>
<td>Specify a PSB BMP for the Online Reorg function.</td>
</tr>
<tr>
<td>REPORTS</td>
<td>See page 526</td>
<td>See page 526</td>
<td>526</td>
<td>Generate reports.</td>
</tr>
<tr>
<td>RMBYTES</td>
<td>(SEGSIZE) (MINBYTES)</td>
<td>(MINBYTES)</td>
<td>529</td>
<td>Specify how to calculate the database record size to determine when to place consecutively inserted segments into overflow.</td>
</tr>
<tr>
<td>ROUTCODE</td>
<td>(nn, ...)</td>
<td>(2,7)</td>
<td>490</td>
<td>Specify routing codes to control the destination of messages sent to the console.</td>
</tr>
<tr>
<td>SCAN</td>
<td>(-nnn+mmm), (+nnn-mmm), (+-nnn), (+-nnn)</td>
<td>(0) for HIDAM (-10+10) for HDAM</td>
<td>531</td>
<td>Set the space search method.</td>
</tr>
<tr>
<td>SEGPOS</td>
<td>(segname1,n1, segname2,n2,...)</td>
<td>none</td>
<td>535</td>
<td>Specify which segment type to place near the root.</td>
</tr>
<tr>
<td>SEGPOSO</td>
<td>(ALL) (segname1, segname2,...)</td>
<td>none</td>
<td>536</td>
<td>Place the specified segment in the HDAM overflow area if the segment exceeds the limit specified in SEGPOS.</td>
</tr>
<tr>
<td>SHARE</td>
<td>(YES,UPDATE)</td>
<td>(NO)</td>
<td>540</td>
<td>Specify whether users will have access to the database during reorganization.</td>
</tr>
<tr>
<td>SIUPRINT</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>522</td>
<td>Specify the data set that will contain the Index Build function reports and processing output.</td>
</tr>
<tr>
<td>SNAPSHOT</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>543</td>
<td>Specify whether to take a Snapshot Copy of the database being unloaded.</td>
</tr>
<tr>
<td>SORTWORK</td>
<td>(n,ccc,unitName dataclas, mgmtclas, storclas)</td>
<td>(3,100,SYSDA)</td>
<td>545</td>
<td>Dynamically allocate sort work data sets.</td>
</tr>
<tr>
<td>SSID</td>
<td>(imsid)</td>
<td>Use the IMSID from IMS generation (DF5VC000 module).</td>
<td>547</td>
<td>Specify the IMS subsystem to be used for the BMP to connect to the IMS control region.</td>
</tr>
<tr>
<td>SWAP</td>
<td>(YES) (NO) (DELETE)</td>
<td>(NO)</td>
<td>548</td>
<td>Automatically swap the data set names after a successful reorganization.</td>
</tr>
</tbody>
</table>
Table 9  Online Reorg function keywords (part 5 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRUNC</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>556</td>
<td>Start each HIDAM root in a new block.</td>
</tr>
<tr>
<td>ULPPRINT</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>522</td>
<td>Specify the data set that will contain the Unload function reports and processing output.</td>
</tr>
<tr>
<td>USEREXIT</td>
<td>(exitName,U</td>
<td>L,YES</td>
<td>NO)</td>
<td>none</td>
</tr>
<tr>
<td>XBMSSID</td>
<td>(ssid)</td>
<td>none</td>
<td>574</td>
<td>Specify the XBM subsystem to be used with the Snapshot Copy function.</td>
</tr>
</tbody>
</table>

PLUSIN examples

The following examples show some possible combinations of keywords. In each example, the Online Reorg function is invoked.

**Example 1:**
The data set names are swapped automatically after the reorganization is complete:

```
REORG SHARE(YES,UPDATE) SWAP(YES)
```

**Example 2:**
The files IDCIN and IDCOUT are used for IDCAMS processing of the shadow data sets:

```
REORG SHARE(YES,UPDATE) IDCAMS(IDCIN,IDCOUT)
```

**Example 3:**
The data set names are not swapped automatically. The RESTART command must be executed in a separate job step to swap the data set names:

```
REORG SHARE(YES,UPDATE) SWAP(NO)
```
Global options module

The global options module for the Online Reorg function is different than the global options modules for MAXM Reorg/EP solutions and the MAXM Reorg/EP Express solution.

To create a global options module for the Online Reorg function, or to change the options set in the global options module, execute the environment setup member (DLIGSET0). The job is in member #DBUGLBL of the sample library. This member documents which options you can change.

The Online Reorg function also uses the global options module that you created for the Reorg function (FRF$GLBL). DLIGSET0 contains options specific to the Online Reorg function, and FRF$GLBL contains options specific to the Reorg function.

No ISPF interface is available for the Online Reorg function.

Report

The Online Reorg function generates the report shown in Table 10. When you reorganize a database with MAXM Reorg/Online, one report is generated for the Online Reorg function (in addition to the reports that are generated by the other functions).

Table 10  Online Reorg function report

<table>
<thead>
<tr>
<th>Report</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concurrent Reorganization</td>
<td>586</td>
</tr>
</tbody>
</table>

Because the Online Reorg function uses the DI+ DBRC interface, the following DI+ messages are issued during an Online Reorg function job, regardless of whether you have DI+ installed:

- BMC44001I
- BMC44008I
- BMC44009I
Unload phase

When the database is copied, the Online Reorg function applies the logs to the copy so that the shadow database is identical to the original database.

You can create a copy of the database to use for other reasons. For more information, see Chapter 9, “Copy function.”

When the shadow database is identical to the original database, the Reorg function manages the reorganization. The Unload function unloads the database and writes it to temporary files. The temporary file is on DASD. No JCL changes are required; MAXM Reorg/Online allocates the temporary file if a DFSURGU2 DD statement is not specified. When the unload is complete, the shadow data sets are deleted and defined (if applicable). The Reload function reads the temporary file and reloads the records directly to the shadow data sets.

Post-processing phase

Post-processing is performed after the database has been reorganized but before the database is ready for access by users. Because the database has been reorganized, the post-processing tasks must be completed. During post-processing, messages are issued that report the beginning and completion of each task. These messages are written to the BMCMSG data set. You can delay post-processing tasks; however, if you do, you must execute the restart process to complete them.

Post-processing includes the following tasks:

- restart processing, if needed
- data set name swapping
- DBRC notifications

For more information, see Appendix B, “Post-processing.”

**NOTE**

If for any reason the job does not complete the post-processing tasks, you must execute the restart process or perform the post-processing tasks manually.

If you specify SWAP(NO) to delay the post-processing tasks, you must execute the restart process to swap the data set names. For more information, see “Restart process” on page 631.
Automated operator interface

The Online Reorg function uses an automated operator interface (AOI) at the beginning of the reorganization and during the data set name swapping task. Online Reorg uses internal logic to communicate with IMS.

Online Reorg issues a /DIS DB dbdName command to check the status of the database. Online Reorg analyzes the response of the /DIS DB command. The AOI proceeds as follows, based on the status of the database:

- **STOPPED,NOTOPEN**
  
  If the database is stopped and not open, Online Reorg does not need to issue any commands. The database can enter the reorganization phase.

- any other status

  Online Reorg issues a /BMCPAU command to quiesce the database.

If the database is in any status other than STOPPED,NOTOPEN, the following process occurs:

1. The AOI replaces the first two /DBR commands in the online reorganization process with the /BMCPAU command.

   This command establishes a sync point instead of taking the database offline. Online Reorg sets a quiesce bit in the IMS control blocks of the database. This bit tells the scheduler to stop scheduling transactions that contain the database in its PSB.

2. Online Reorg waits five seconds and scans all transactions to see whether any active transactions are accessing the database.

   If active transactions exist, Online Reorg waits another five seconds and checks again. This cycle repeats until the transaction activity for the database has quiesced or 30 attempts to quiesce have failed.

   **NOTE**

   If Online Reorg fails to obtain a sync point, the function automatically switches to /DBR processing.
3. When all transaction activity has finished and no active transactions are using the database, the CRF component takes one of the following actions (based on whether the sync point occurred before or after the copy phase):

- turns on Database Log Capture
- turns off Database Log Capture and turns on Database Change Capture

4. Online Reorg issues a /STA command on the database, which reschedules any suspended transactions.

**NOTE**

If SNAPSHOT is being used, Online Reorg sets a single sync point at the beginning of the reorganization. In this case, only Database Change Capture is turned on. /BMCPAU is not used with SNAPSHOT, so Online Reorg issues a /DBR command to establish the sync point.

If necessary, the AOI issues a /DBR command. Online Reorg waits five seconds, issues a /DIS DB command, and checks the response. If any database does not have STOPPED,NOTOPEN status, Online Reorg issues another /DIS DB command. This cycle repeats until the database is stopped or 30 attempts have failed. If any databases are not stopped after 30 attempts, messages BMC90465 and BMC90466 are issued to request operator intervention.

---

**BMPs**

If you are using the AR/CTL product to manage BMPs, the Online Reorg function communicates with AR/CTL in the following ways:

- tells AR/CTL to suspend BMP processing before a /BMCPAU or /DBR command
- tells AR/CTL to resume BMP processing after a /STA command

AR/CTL suspends and resumes BMP processing without stopping the job step. It appears to MVS that the job is still running. You do not need to resubmit a job step to get the BMP running again. When Online Reorg checks the status of the database that will be reorganized, it might find that the database has BMPs running against it. If Online Reorg finds any BMPs running, message BMC90486I is issued to report which BMPs are running on which IMS systems.

To use the AR/CTL interface, include the BMC load library in the STEPLIB concatenation of the Online Reorg job. You do not need to include any keywords in the PLUSIN control statement, and you do not need to include an ARCSYSIN DD statement.

For more information about AR/CTL, see the *Application Enhancement Solutions Administrator Guide*. 
If you have BMPs that are not under the control of AR/CTL, Online Reorg issues a BMC message indicating that the BMP is running and needs to stop.

**WFI BMP processing**

The following example JCL shows how to process WFI BMPs using interrupt and resume processing under CRF:

**Figure 14  WFI BMP processing**

```plaintext
//*****************************************************
//*        EXECUTE WFIBMPE AS WFI BMP
//*****************************************************
//*
//WFIBM01  EXEC PGM=DLIWFI,
//         PARM='BMP,WFIBMPE,WFIBMPE,WFIBMPE,.......,,MXOC'
//*
//STEPLIB  DD  DISP=SHR,DSN=BMCDBU.R47.LOAD
//         DD  DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD  DISP=SHR,DSN=IMSVS.RESLIB
//IMS      DD  DISP=SHR,DSN=IMSVS.DBDLIB
//         DD  DISP=SHR,DSN=IMSVS.PSBLIB
.
.
.
```

**NOTE**

AR/CTL is not required. The JCL is changed to include the DBU code in STEPLIB for the WFI BMP and DLIWFI is substituted for DFSRRC00.

During the Online Reorg, the WFI pauses and resumes automatically. Three pauses take place: before the copy phase, after the copy phase, and during the post-processing phase. Updates are captured and applied at the proper points.

**DBR commands**

The Online Reorg function enables you to reorganize a database and leave it available online to applications. Online Reorg must have exclusive access to the database for a short time at the beginning and end of the reorganization. While Online Reorg has exclusive control, it quiesces and starts the database that is being reorganized.
When you submit the reorganization job, Online Reorg uses its AOI to issue an IMS /BMCPAU or /DBR DB dbdName NOFEOV command to clear all buffers and quiesce the database. At this point, the database is in recording mode and Online Reorg issues a /STA command to get the database back online and to release any suspended transactions. The database is reopened, DBRC authorization is granted, Online Reorg prepares to record the updates, and the reorganization begins. The entire process should take only a few seconds.

When the reorganization is complete and an image copy has been taken, Online Reorg uses a batch job to apply the changes to the reorganized (shadow) database. This is a standard batch update job, and it creates a standard batch log. In most standard batch jobs, a /DBR must occur before a batch update job can begin. After the job is complete, a /STA command is issued to start the database. Between the /DBR and the /STA, the batch update job has exclusive control of the database. This is also true for the Online Reorg function.

In general, when a /DBR is issued, the OLDS are switched. Online Reorg allows you to not switch the logs when the /DBR is issued (see “DBR” on page 421). Online Reorg issues an /SWI OLDS command before starting the databases.

**DBD changes during reorganization**

You can change HDAM randomizer DBD parameters during an Online Reorg function reorganization by specifying the OLDDBD keyword in the reorganization job. When you change DBD values, you are responsible for ensuring that the new values are valid.

If you need to change other DBD parameters, use the Reorg function to reorganize the database. The Reorg function allows you to change certain DBD parameters when you use the OLDDBD keyword. For more information, see “OLDDBD” on page 496.

You cannot change the physical characteristics of the database (BLKSIZE or CISIZE) during the reorganization.

Be careful when using the OLDDBD and SWAP(YES) keywords during an Online Reorg function reorganization; changing the DBD will affect your IMS online systems. The effect depends on whether you have the BMC DELTA IMS® DB/DC product installed.
DELTA IMS DB/DC installed

The DELTA IMS DB/DC interface to the Online Reorg function automates the process of reorganizing a database when you use the OLDDBD keyword and specify SWAP(YES). Before you execute the reorganization job step, you must have a BMCLINK started task running to communicate with DELTA IMS DB/DC in each online IMS system. You must also ensure that you set DELTA IMS DB/DC options to use VTAM to communicate with the online system. For more information about setting DELTA IMS DB/DC options, see the DELTA IMS User Guide.

Make the following changes to the Online Reorg job step JCL:

1 Include the DELTA IMS DB/DC load library in the STEPLIB.

2 Include an ACBSYSIN DD statement that lists all sets of staging source (by IMS release), ACBLIBA, ACBLIBB, and MODSTATS data sets that contain the DMBs for the database being reorganized. If the ACBSYSIN DD statement is not included or if it is specified as DUMMY, the DELTA IMS DB/DC interface to the Online Reorg function will not execute. If the ACBSYSIN DD statement is included but is empty, error message BMC90308C is issued.

3 Include an ACBLIB as the source for each set of libraries, for each level of IMS that is shared. This ACBLIB must contain new DMBs for all DBDs used in the reorganization job step. The DMBs must match the DBDs named in the reorganization job step IMS DD statement that contains the new DBD.

By default, Online Reorg copies the DMB from ACBLIB to the active online ACBLIBx. To have the DMBs copied into both the active and inactive ACBLIBx libraries, specify BOTH in the ACBSYSIN DD data set. Whether you specify ACTIVE or BOTH, the target ACBLIBx libraries are searched to ensure that they contain at least one of the DMBs. If the ACBLIBx libraries do not contain at least one of the DMBs, the DMBs are not copied to that online ACBLIBx library. For concatenated ACBLIBx data sets, all occurrences of a change DMB will be replaced.

DELTA IMS DB/DC forces the online IMS systems to load the new DMB. Because this process can also be executed during the restart process, do not copy the new DBD to the OLDDBD library before the post-processing tasks are restarted.

**NOTE**

BMC recommends that you have DI+ installed when you use this interface. DI+ ensures that you are using the correct ACB.
Figure 15 shows example JCL.

Figure 15  JCL for OLDDBD with DELTA IMS DB/DC

```
//ACBSYSIN DD *
  [BOTH | ACTIVE]
  ACBLIB(your.staging.aclib[,n]) -
  ACBLIBA(your.first.acliba,your.second.acliba,...) -
  ACBLIBB(your.first.aclibb,your.second.aclibb,...) -
  MODSTAT(your.ims.modstat)  <=optional with BOTH
  ACBLIB(your.staging.aclib[,n]) -
  ACBLIBA(other.first.acliba,other.second.acliba,...) -
  ACBLIBB(other.first.aclibb,other.second.aclibb,...) -
  MODSTAT(other.ims.modstat)  <=optional with BOTH
```

The example includes the following control statements:

Table 11  Control statements for JCL for OLDDBD with DELTA IMS DB/DC

<table>
<thead>
<tr>
<th>Control statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOTH</td>
<td>Specify BOTH to write to ACBLIBA and ACBLIBB.</td>
</tr>
<tr>
<td>ACTIVE</td>
<td>Specify ACTIVE to write to only the active ACBLIBx. ACTIVE is the default.</td>
</tr>
<tr>
<td>ACBLIB</td>
<td>The source of the ACBLIB libraries. n specifies the IMS level of the ACBLIB if it is different than the IMS level of the job.</td>
</tr>
<tr>
<td>ACBLIBA</td>
<td>The concatenation of ACBLIBA libraries in your IMS control region.</td>
</tr>
<tr>
<td>ACBLIBB</td>
<td>The concatenation of ACBLIBB libraries in your IMS control region.</td>
</tr>
<tr>
<td>MODSTAT</td>
<td>The MODSTAT data set for your IMS system. The MODSTAT statement is required when you specify ACTIVE; it is optional when you specify BOTH.</td>
</tr>
</tbody>
</table>

If the DELTA IMS DB/DC interface fails, the Online Reorg function aborts the DELTA IMS phase but continues with post-processing tasks and then abends with U4094 and message BMC1802xx. The DELTA IMS DB/DC interface will fail if any of the following situations occur:

- Incorrect information is specified in the ACBSYSIN control statement data set.
- The new DMBs are not found in the ACBLIB.
- Any ACBLIB errors are found.
- VTAM is recycled but BMCLINK has not been recycled. If VTAM is recycled, BMCLINK must be recycled before it is usable.
If the DELTA IMS DB/DC initialization is successful but Online Reorg detects a problem after the IEBCOPY of the ACBLIB to the active ACBLIBx, Online reorg issues a WTOR. Reply R (retry) or S (skip) to the WTOR. If you reply S (skip), Online Reorg skips the DELTA IMS DB/DC phases, completes post-processing tasks, and abends with U4094 and message BMC1802xx.

If an abend occurs during DELTA IMS DB/DC processing in the Online Reorg job, check the log for message BMC180213C. If this message was not issued, run the restart process (see Appendix B, “Post-processing”). The copy process will be retried during the restart process; however, if message BMC180213C was issued, you must perform the copy tasks manually.

**DELTA IMS DB/DC not installed**

When you use the OLDDBD keyword and DELTA IMS DB/DC is not installed in any of the target IMS online systems, you must complete the following tasks after the reorganization is complete and before the database is restarted for each IMS system that shares the changed DBD:

1. Execute an ACBGEN for the new (changed) DBD after the data set names are swapped.

2. Issue a /MODIFY PREPARE ACBLIB command and a /MODIFY COMMIT command to activate the new ACB.

   Because the database must use the correct ACB, you must ensure that the new ACB has been generated and activated before the next step.

   **WARNING**

   If you do not execute an ACBGEN for the new DBD and issue the appropriate /MODIFY commands before issuing the /STA DB command (the next step), you will create data integrity exposures and damage the database.

3. Issue a /STA DB dbdName GLOBAL command to start the database.

   If you specify SWAP(YES), the Online Reorg function swaps the data set names but leaves the database offline and in NOAUTH status in DBRC.

   **WARNING**

   Online Reorg does not automatically issue a /STA DB dbdName GLOBAL command to turn the NOAUTH status off and release the database for use.
Database recovery

If you need to recover a database that was reorganized by the Online Reorg function, note that the image copy time stamp in the header of the image copy data set created by Online Reorg is not the same as the time stamp posted in DBRC. The time stamp in the image copy header is created at image copy initialization, and the NOTIFY.IC is posted at the end of the reorganization for data integrity purposes. This situation can affect recovery.

RECOVERY PLUS for IMS handles this situation with no manual intervention. However, if you are using the IMS Recovery utility, the utility will issue a message indicating that the time stamps do not match. To avoid zapping the IMS Recovery utility to bypass this check, BMC recommends that you concatenate the BMC load library in your IMS Recovery utility STEPLIB concatenation (ahead of RESLIB) when you recover a database using the image copy created during the Online Reorg function reorganization. Including the BMC load library in your IMS Recovery utility STEPLIB concatenation ahead of RESLIB causes Online Reorg to dynamically bypass checking the time stamps (no usermod or zap is required).

Non-disruptive maintenance

You can update the Change Recording Facility (CRF) and Online Defrag (OLR) components that execute in the IMS control region without requiring that the IMS control region be recycled. For information about implementing this function, see the BMC Products for IMS Installation Guide.
Non-disruptive maintenance
Online/Defrag feature

The Online/Defrag function is a BMC signature innovation that offers customized analyses so that you can identify and reorganize only those areas of a database that need to be reorganized. Like the Online Reorg function, the Online/Defrag function allows you stay online during a database reorganization.

This chapter includes the following topics:

Overview ................................................................. 135
  Record mode process .............................................. 137
  Scan mode process ................................................. 139
  Block Mode Process ................................................ 139
Online/Defrag options .................................................. 140
Online/Defrag options .................................................. 140
  Extractor and Recommendation Selector ....................... 146
  Analyzer ............................................................... 150
Online reorganizations .................................................. 153
  System flow .......................................................... 153
  JCL requirements ..................................................... 154
  Examples .............................................................. 156
Options ................................................................. 159
Reports ................................................................. 159

Overview

No function has been able to reorganize a database while the database remains online and available for updates—*until now.*
Online/Defrag provides increased availability and performance for IMS databases that require continuous or near-continuous availability. Online/Defrag can reorganize a single database record or an entire database with

- no down time
- minimal user interaction
- concurrent updates

Online/Defrag provides reorganization functions for HDAM, HIDAM, PHDAM, and PHIDAM databases that are experiencing performance problems caused by record fragmentation. Online/Defrag offers an analysis component that lets you target a reorganization to only those areas within the database that need to be reorganized. Moreover, Online/Defrag recommends the reorganization parameters that will reduce I/O activity the most.

**NOTE**

When Online/Defrag is used on PHDAM or PHIDAM databases, it reorganizes one partition each time it is executed.

Online/Defrag does not replace existing BMC reorganization utilities because those utilities are still needed for reorganizations that require database description (DBD) or space allocation changes. Online/Defrag complements the existing reorganization utilities to provide a comprehensive end-to-end reorganization solution for IMS full-function databases.

The Online/Defrag job schedules a batch message processing (BMP) program to communicate with DBCTL. The process by which the database is reorganized is dependent on the specified mode. Online/Defrag offers three reorganization modes:

- **Record**: Online/Defrag reorganizes only those records that are recommended for reorganization by the Online/Defrag Record Analysis.

- **Scan**: Online/Defrag analyzes every record in the database and reorganizes only those records that meet specified criteria.

- **Block**: Online/Defrag reorganizes every record in the database. No analysis is performed to determine the fragmentation level of the records.

During initialization, Online/Defrag identifies blocks that have a user-defined amount of free space. Blocks are identified as free or full. The free blocks are reserved for use as reorganization targets. There must be enough free space in the database to reorganize the largest record. A record might be skipped if not enough target blocks are available to contain it.
Block status (free or full) is continuously updated throughout the reorganization process. Blocks that were previously free might become full; blocks that were previously full become free when they meet the user-defined amount of free space. The status is dynamic because of the movement of records during the reorganization.

Online/Defrag lets you interrupt a reorganization and then resume the same job. When the job is interrupted, Online/Defrag stores the key value of the restart record for use when the reorganization is restarted. For more information about interrupting and resuming a reorganization, see “RESUME” on page 528.

Record mode process

Record mode reorganization is an exclusive BMC technology that lets you perform a targeted, online reorganization. The process for record mode reorganizations involves the following functions:

1. Record Analysis

   The analysis is an independent component of Online/Defrag that precedes the online reorganization. The Online/Defrag Record Analysis identifies records that are causing performance problems due to space fragmentation and generates a recommendation report. Online/Defrag only reorganizes the records that are recommended for reorganization.

2. Free Space Determination

   Online/Defrag identifies blocks that have a user-defined amount of free space. The free blocks are reserved for use as reorganization targets. Block status is updated continuously throughout the reorganization process.

3. Online Reorganization

   After free space is identified, the Online/Defrag job accepts input (the Record Analysis recommendation) that identifies specific records by key. The Online/Defrag job reanalyzes the records to verify that they continue to meet reorganization parameters and then reorganizes the records into the target blocks.

4. Reports Review

   Online/Defrag generates reports that summarize reorganization activities. For record mode reorganizations, detailed information is stored in the job message output.

For detailed information about the record mode process, refer to “Record mode JCL” on page 156.
Figure 16 shows a record that spans six database blocks. Analysis of the database record indicates that the minimum number of blocks that the record requires is three. If reorganization parameters specify an expected block improvement of 50 percent, the Online/Defrag Record Analysis will recommend record 1 for reorganization.

Figure 16  Record 1 before record mode reorganization

---

Figure 17 shows record 1 after reorganization. Record 1 was relocated to three adjacent target blocks. The expected block improvement of 50 percent was satisfied. Record 1 spans half the number of blocks that it spanned before reorganization. Accessing record 1 after the reorganization requires fewer I/Os, which increases database performance.

Figure 17  Record 1 after record mode reorganization

---

Some database records might have more activity (updates) than other records. The greater the activity, the greater the disorganization. Identifying and reorganizing only those records that are substantially disorganized maximizes efficiency of database reorganization and improves overall database performance.

**NOTE**

Record mode reorganization is best for databases in which applications use random access to the data.
Scan mode process

Scan mode reorganization is similar to record mode reorganization. Online/Defrag scans the entire database, analyzes every record, and reorganizes the records that meet user-defined criteria. The difference between the two modes is that a scan reorganization does not take advantage of the recommendation report generated by the Online/Defrag Record Analysis. Instead, Online/Defrag analyzes every record during the online reorganization process. The process for scan mode reorganizations involves the following functions:

1. Free Space Determination

Online/Defrag identifies blocks that have a user-defined amount of free space. The free blocks are reserved for use as reorganization targets. Block status is continuously updated throughout the reorganization process.

2. Online Reorganization

After free space is identified, the Online/Defrag job analyzes every record in the database. If the record meets reorganization parameters, it is reorganized into the target blocks.

3. Reports Review

Online/Defrag generates reports that summarize reorganization activities. The reports contain information such as the percentage of block improvement after the reorganization. When scan mode is specified, detailed messages about each record reorganization are not automatically generated because of the potential size of the report. However, you can use an Online/Defrag keyword to specify the option to generate record details.

Block Mode Process

Block mode reorganization emulates a traditional reorganization without the associated outage or shadow data sets. When block mode is specified, each record is reorganized into the target blocks regardless of its fragmentation level. The process for block mode reorganizations involves the following functions:

1. Free Space Determination

Online/Defrag identifies blocks that have a user-defined amount of free space. The free blocks are reserved for use as reorganization targets. Block status is continuously updated throughout the reorganization process.
2. Online Reorganization

After free space is identified, the Online/Defrag job reorganizes every record in the database into the target blocks. Records are reorganized regardless of their fragmentation level.

3. Reports Review

Online/Defrag generates reports that summarize reorganization activities. The reports contain information such as detailed record reorganization messages and the percentage of block improvement after the reorganization.

---

**NOTE**
To make any DBD or space allocation changes, you must use traditional reorganization utilities to reorganize the database.

---

**Online/Defrag options**

Online/Defrag follows an iterative process to determine the option values to use for a particular database. Each succeeding step adds to or replaces option values determined in the previous step:

1. Online/Defrag builds an initial parameter table from the default values that are coded internally. The only values that are coded internally are for extracting data during the Record Analysis.

2. Online/Defrag searches the Online/Defrag load module library for a global options module named DLIMPARM. If Online/Defrag finds the DLIMPARM module, it uses the values coded in that module to replace values in the initial parameter table.

3. Online/Defrag uses keyword values that are coded in the control statement to replace the table options which are built in the previous steps.

Figure 18 on page 141 represents the parameter table that Online/Defrag uses to build options during the reorganization of the database.
Table 12 lists and describes Online/Defrag commands.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALYZE</td>
<td>Invoke the Extractor and Recommendation Selector, and the Online/Defrag Analyzer when MODE(RECORD) is specified.</td>
</tr>
<tr>
<td>OREORG</td>
<td>Perform a reorganization on the input data set. Online/Defrag does not support logical relationships.</td>
</tr>
</tbody>
</table>

Table 13 lists and describes the keywords that you can specify in the PLUSIN control statement. Refer to the corresponding page number for more information about each keyword.

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGN</td>
<td>OREORG</td>
<td>(agnName)</td>
<td>none</td>
<td>Specify the application group name for the BMP subtask.</td>
<td>407</td>
</tr>
<tr>
<td>AUTOSTOP</td>
<td>OREORG</td>
<td>(TIME,hhmm)</td>
<td>none</td>
<td>Specify parameters to automatically stop the online reorganization.</td>
<td>411</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(MINUTES,n)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(RECORDS,n)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(KEY,keyValue)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIPF</td>
<td>OREORG ANALYZE</td>
<td>1-99</td>
<td>(25)</td>
<td>Specify the expected percentage of block improvement to qualify a record for reorganization.</td>
<td>413</td>
</tr>
</tbody>
</table>
Table 13  Online/Defrag options (part 2 of 3)

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>OREORG</td>
<td>(dbname)</td>
<td>none</td>
<td>Specify the DBD to be reorganized or analyzed.</td>
<td>420</td>
</tr>
<tr>
<td></td>
<td>ANALYZE</td>
<td></td>
<td>the DBD processed by the Extractor (ANALYZE command only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DESC_CODE</td>
<td>OREORG</td>
<td>(nn,...)</td>
<td>(7)</td>
<td>Specify descriptor codes used to route messages to the console.</td>
<td>490</td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>OREORG</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>Specify dynamic allocation of database data sets.</td>
<td>430</td>
</tr>
<tr>
<td>FSPF</td>
<td>OREORG</td>
<td>(1-100)</td>
<td>(95)</td>
<td>Specify the percentage of free space to qualify a block as free.</td>
<td>441</td>
</tr>
<tr>
<td>KEYS</td>
<td>OREORG</td>
<td>(rootKey)</td>
<td>none</td>
<td>Specify the root key value of records to be reorganized or analyzed.</td>
<td>478</td>
</tr>
<tr>
<td>MBI</td>
<td>OREORG</td>
<td>(1-9999)</td>
<td>(5)</td>
<td>Specify the minimum number of expected blocks to be saved per record.</td>
<td>488</td>
</tr>
<tr>
<td>MINREC</td>
<td>OREORG</td>
<td>(1-9,999,999)</td>
<td>variable 20% of available records</td>
<td>Specify the minimum number of records to reorganize or analyze.</td>
<td>489</td>
</tr>
<tr>
<td>MODE</td>
<td>OREORG</td>
<td>(RECORD) (SCAN) (BLOCK)</td>
<td>none</td>
<td>Specify the mode of reorganization.</td>
<td>489</td>
</tr>
<tr>
<td>MONITOR</td>
<td>OREORG</td>
<td>(nnnnnnn)</td>
<td>no automatic monitoring</td>
<td>Monitor job progress.</td>
<td>490</td>
</tr>
<tr>
<td>MONUSERS</td>
<td>OREORG</td>
<td>(user01, user02,...)</td>
<td>Send monitor messages to TSO users and BMCMSG.</td>
<td>Send monitor messages to up to 10 specified TSO user IDs.</td>
<td>490</td>
</tr>
<tr>
<td>OLRPRINT</td>
<td>OREORG</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>Specify the data set to contain reports and other output.</td>
<td>522</td>
</tr>
<tr>
<td>PART</td>
<td>OREORG</td>
<td>(partNum)</td>
<td>none</td>
<td>Specify the partition to reorganize.</td>
<td>508</td>
</tr>
<tr>
<td>PDX</td>
<td>OREORG</td>
<td>(OKAYnnnn) (STOPnnnn)</td>
<td>(OKAY0004)</td>
<td>Specify the action to take if PDX is unavailable.</td>
<td>519</td>
</tr>
</tbody>
</table>
The Online/Defrag Extractor offers a global options module. DLIMPARM is a load module that is assembled and link-edited during installation using the $OLRPARM macro. To set or modify default values, you must edit and reassemble DLIMPARM.

The $OLRPARM macro also includes the following options to specify for the Online/Defrag reorganization job:

- **PDXDSN**
- **PDX**

The PDXDSN option specifies the name of the PDX data set. The PDX data set is used to store a record that contains information for use when a reorganization is interrupted and later resumed. The PDXDSN value can be specified as a global option or in the //PDX DD statement. The JCL value overrides the global option value.

The PDX keyword specifies the action to take if PDXDSN is unavailable. The PDX value can be specified as a global option in $OLRPARM or in the PLUSIN control statement.
Online/Defrag Record Analysis

The Record Analysis is an independent component of Online/Defrag that precedes a record mode reorganization.

1 Online/Defrag Extractor and Recommendation Selector

The Online/Defrag Extractor and Recommendation Selector process reads the database and produces files that are used to maximize record or scan mode reorganizations. The process calculates parameters that will result in the greatest reduction of I/O activity (buffer searches).

2 Online/Defrag Analyzer

The Online/Defrag Analyzer receives the output data sets from the Online/Defrag Extractor and Recommendation Selector process and analyzes each record in the data set to determine candidacy for reorganization.

NOTE

The Online/Defrag job analyzes the records identified in the key list data set again to verify that they continue to meet reorganization criteria. The Online/Defrag job may specify reorganization criteria that is more or less stringent than the criteria specified during the Record Analysis.

Figure 19 shows the system flow for the Record Analysis.
The Online/Defrag Record Analysis uses the following components and processes:

### Table 14  Components and processes of the Online/Defrag Record Analysis

<table>
<thead>
<tr>
<th>Component or process</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online/Defrag Extractor and Recommendation Selector</td>
<td>The Extractor gathers information required to analyze database records and produces an output data set. The Recommendation Selector uses the Extractor output and the MINREC keyword to calculate the combination of MBI and BIPF values that will result in the greatest reduction of I/O activity. The administrator defines the minimum number of records to be reorganized. The Recommendation Selector creates three output data sets: reports, an output data set directed to Online/Defrag Analyzer (ANALYZE command), and an output data set directed to Online/Defrag (OREORG command).</td>
</tr>
<tr>
<td>Output data set, OREORG</td>
<td>The output data set with the OREORG command is input to the Online/Defrag function.</td>
</tr>
<tr>
<td>Output data set, ANALYZE</td>
<td>The output data set with the ANALYZE command is input to the Online/Defrag Analyzer for record mode reorganizations. The Online/Defrag Analyzer verifies that the Recommendation Selector values for the MBI and BIPF keywords will result in the greatest reduction of I/O activity.</td>
</tr>
<tr>
<td>Online/Defrag Analyzer</td>
<td>Online/Defrag Analyzer uses the DBD, BIPF, MBI, and FSPF keywords to identify records that are candidates for reorganization. Online/Defrag Analyzer produces the key list data set that is passed to the Online/Defrag job.</td>
</tr>
<tr>
<td>Key list data set</td>
<td>The key list data set is the table of record keys that Online/Defrag Analyzer generates when MODE(RECORD) is specified. The table identifies records that are candidates for reorganization. The key list data set can be used as input to the Online/Defrag job.</td>
</tr>
<tr>
<td>Online/Defrag</td>
<td>Online/Defrag is the online reorganization job.</td>
</tr>
<tr>
<td>Keywords</td>
<td>Keywords control the extraction, recommendation selection, analysis, and reorganization.</td>
</tr>
<tr>
<td>Reports</td>
<td>Online/Defrag Extractor and Recommendation Selector, Online/Defrag Analyzer, and Online/Defrag automatically generate reports.</td>
</tr>
</tbody>
</table>
Extractor and Recommendation Selector

The Online/Defrag Extractor and Recommendation Selector process reads the database and produces files that are used to maximize record or scan mode reorganizations. The process calculates parameters that will result in the greatest reduction of I/O activity (buffer searches).

JCL requirements

Figure 20 shows an example of Online/Defrag Extractor and Recommendation Selector JCL.

Figure 20  Online/Defrag Extractor and Recommendation Selector JCL

A summary of the JCL statements to run the Online/Defrag Extractor and Recommendation Selector follows:

EXEC

Required. You must specify the DLIAANL2 program to execute the Online/Defrag Extractor and Recommendation Selector.

STEPLIB DD

Optional. Defines the library that contains the Online/Defrag modules. The STEPLIB concatenation must contain an MDALIB for Online/Defrag to access if the RECON DD statements are not in the JCL. Always concatenate the BMC library before the IMS library.
Extractor and Recommendation Selector

**DFSRESLIB DD**

Required. Defines the library containing the RESLIB.

**IMS DD**

Required. Defines the library, typically IMSVS.DBDLIB, containing the DBD referenced in the Online/Defrag control statements.

**RECONx DD**

Required. Defines the DBRC data sets. An alternative is to specify an MDALIB (that contains RECON data sets) to your STEPLIB concatenation.

**BMCMSG DD**

Optional. Defines the output data set that contains messages and a listing of PLUSIN control statements. If the BMCMSG DD statement is not included in the JCL, Online/Defrag dynamically allocates a BMCMSG data set.

**ANLPRINT DD**

Required. Defines the data sets that contain reports generated by the Online/Defrag Analyzer.

**HSRCNTLI DD**

Required. An 80-byte sequential file that contains the recommended control cards for Online/Defrag (BMCOLR). The HSRCNTLI DD statement must be edited to include the SSID keyword. The alternative is to concatenate the HSRCNTLI DD statement with other keywords in subsequent data sets.

**DMNCNTLI DD**

Required. An 80-byte sequential file that contains the recommended control cards for the Online/Defrag Analyzer (DLIMDMAN). You may specify DMNCNTLI without modification as the PLUSIN DD statement for the Online/Defrag Analyzer.

**SYSUDUMP DD**

Optional. Defines a print data set to contain a system dump if an abend occurs during the job step.
**EXTRACTO DD**

Required for record mode reorganizations. Defines the data set extracted by Online/Defrag. The EXTRACTO DD statement is renamed PCPDANIP when processed by the DLIMDMAN step.

**DATABASE DD**

Required. Defines the database that is being analyzed.

**PLUSIN DD**

Required. Defines user options. Options include the ANALYZE command and keywords that specify the parameters of the recommendation analysis. If you do not specify parameters, default values are used.

**Options**

Table 15 shows the keywords used for the Online/Defrag Extractor and Recommendation Selector. Refer to the corresponding page number for more information about the keyword.

<table>
<thead>
<tr>
<th>Option</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>ANALYZE</td>
<td>(dbdName) not required with ANALYZE</td>
<td>The name of the DBD processed by the Extractor.</td>
<td>Specify the DBD to analyze.</td>
<td>420</td>
</tr>
<tr>
<td>FSPF</td>
<td>ANALYZE</td>
<td>(1-100)</td>
<td>(95)</td>
<td>Specify the percentage of free space to qualify a block as free.</td>
<td>441</td>
</tr>
<tr>
<td>MINREC</td>
<td>ANALYZE</td>
<td>(1-9,999,999)</td>
<td>variable (20% of the records that qualify for reorganization)</td>
<td>Specify the minimum number of records to reorganize.</td>
<td>489</td>
</tr>
</tbody>
</table>

To set or modify the default values, you must edit and reassemble the global options module, DLIMPARM.
Reports

Online/Defrag Extractor automatically generates a message log and a Database Analysis Reorganization Recommendation report. The message log is generated automatically in the BMCMMSG data set. The reorganization recommendation report provides the recommended values for the BIPF and MBI keywords that will result in the greatest reduction of I/O activity. The report also provides reorganization result estimates.

Table 16 describes the reorganization recommendation report fields.

Table 16 Database Analysis Reorganization Recommendation report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBDNAME</td>
<td>The name of the database that was analyzed.</td>
</tr>
<tr>
<td>MINREC</td>
<td>The minimum number of records to move during the reorganization. The administrator specifies the MINREC value prior to the Recommendation Selector execution. If the administrator does not specify a value, the default is used.</td>
</tr>
<tr>
<td>BIPF</td>
<td>The BIPF value recommended by the Recommendation Selector.</td>
</tr>
<tr>
<td>MBI</td>
<td>The MBI value recommended by the Recommendation Selector.</td>
</tr>
<tr>
<td>EST I/O SAVED</td>
<td>The estimated reduction of I/O events (buffer searches) after the reorganization.</td>
</tr>
<tr>
<td>TOTAL DBR</td>
<td>The total number of database records in the specified database.</td>
</tr>
<tr>
<td>TOTAL DB SIZE</td>
<td>The total number of bytes that comprise all database records in the specified database.</td>
</tr>
<tr>
<td>QUALIFIED DBR CNT</td>
<td>The number of fragmented database records.</td>
</tr>
<tr>
<td>QUALIFIED DBR SIZE</td>
<td>The number of bytes that comprise all fragmented database records.</td>
</tr>
<tr>
<td>COUNT DBR MOVED</td>
<td>The estimated number of database records that will be moved during the reorganization.</td>
</tr>
<tr>
<td>% OF DBR</td>
<td>The COUNT DBR MOVED value represented as a percent of TOTAL DBR.</td>
</tr>
<tr>
<td>%Q DBR</td>
<td>The COUNT DBR MOVED value represented as a percent of QUALIFIED DBR CNT.</td>
</tr>
<tr>
<td>REORG DATA SIZE</td>
<td>The estimated number of bytes that comprise all database records that will be moved during the reorganization.</td>
</tr>
<tr>
<td>% DB SIZE</td>
<td>The REORG DATA SIZE value represented as a percent of DB SIZE.</td>
</tr>
<tr>
<td>%Q DB SIZE</td>
<td>The REORG DATA SIZE value represented as a percent of QUALIFIED DB SIZE.</td>
</tr>
</tbody>
</table>
The Online/Defrag Analyzer receives the output data sets from the Online/Defrag Extractor and Recommendation Selector process and analyzes each record in the data set to determine candidacy for reorganization.

**JCL requirements**

Figure 21 shows an example of Online/Defrag Analyzer JCL.

**Figure 21  Online/Defrag Analyzer JCL**

```plaintext
EXEC //EXECUTE ONLINE REORG ANALYSIS
HSRAN EXEC PGM=DLIMDMAN
/STEPLIB DD DISP=SHR,DSN=load.dataSet <= OLR LIBRARY
/DFSRESLB DD DISP=SHR,DSN=dfs.dataSet
/IMS DD DISP=SHR,DSN=ims.dataSet
/BMCMSG DD SYSOUT=* 
/ANLPRINT DD SYSOUT=* 
/PCPDANP DD DISP=SHR,DSN=DBU.HSR.EXTRACTO <= EXTRACTO DD FROM DLIAANL2
/ANLPKEYO DD DISP=SHR,DSN=DBU.HSR.KEYLIST <= KEYS FILE FOR OLR
/*
/PLUSIN DD DISP=SHR,DSN=DBU.HSR.DMNCNTLI <= FROM DLIAANL2
/*
```

A summary of the JCL specified for a Record Analysis follows:

**EXEC**

Required. The program name is DLIMDMAN. The region size requirement is a direct function of the buffer space that you request.

**STEPLIB DD**

Optional. Defines the library that contains the Online/Defrag modules. The STEPLIB concatenation must contain an MDALIB for Online/Defrag to access if the RECON DD statements are not in the JCL. Always concatenate the BMC library before the IMS library.

**DFSRESLB DD**

Required. Defines the library containing the RESLIB.
**IMS DD**

Required. Defines the library, typically IMSVS.DBDLIB, that contains the DBD referenced in the Online/Defrag control statements.

**BMCMSG DD**

Optional. Defines the output data set that contains messages and a listing of PLUSIN control statements. If the BMCMSG DD statement is not included in the JCL, Online/Defrag dynamically allocates a BMCMSG data set.

**BMCTRACE DD**

Optional. Defines the BMCTRACE DD statement to record diagnostic information about the job. For more information about BMCTRACE and other diagnostic reports, see Appendix A, “Diagnostic procedures.”

**BMCPRINT DD**

Optional. Defines the output data set that contains reports. If the BMCPRINT DD statement is not included in the JCL, Online/Defrag dynamically allocates a BMCPRINT data set.

**SYSUDUMP DD**

Optional. Defines a print data set to contain a system dump if an abend occurs during the job step.

**ANLPRINT DD**

Optional. Defines a print data set to contain reports generated by the Online/Defrag Analyzer.

**PCPDANIP DD**

Required. Defines the input data set for Online/Defrag Analyzer.

**ANLPKEYO DD**

Required. Defines the output data set from Online/Defrag Analyzer. The ANLPKEYO data set contains the list of keys identifying the records that are recommended for reorganization. The ANLPKEYO DD statement is renamed OLRKEYS when processed by the Online/Defrag job.
Conditional. Defines user options. Options include the ANALYZE command and keywords that specify the parameters of the analysis. If you do not specify reorganization parameters, default values are used.

**Options**

For information about the keywords to use with the Online/Defrag Analyzer, see Table 13 on page 141.

**Reports**

Online/Defrag Analyzer automatically generates a message log and recommendation report. Online/Defrag The message log is generated automatically in the BMCMMSG data set. The recommendation report provides a summary of the Record Analysis. Table 17 describes the report fields.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBDNAME</td>
<td>The name of the database that was analyzed.</td>
</tr>
<tr>
<td>TYPE</td>
<td>The type of record. Online/Defrag supports only standard database records and no twin chain records.</td>
</tr>
<tr>
<td>OLDB</td>
<td>The number of blocks in the database that the record spans.</td>
</tr>
<tr>
<td>NEWB</td>
<td>The number of blocks that the record is expected to span after reorganization.</td>
</tr>
<tr>
<td>LEVEL</td>
<td>The percentage of block improvement that is expected to occur after reorganization. The value is a tentative calculation based on the Record Analysis.</td>
</tr>
<tr>
<td>COUNT</td>
<td>The number of records that meet the reorganization parameters. The parameters include the BIPF, MBI, and FSPF keyword values specified with the ANALYZE command.</td>
</tr>
</tbody>
</table>
Online reorganizations

This section presents the following topics for online reorganizations: system flow, JCL requirements, reorganization examples, user-controlled options, and reports.

System flow

The Online/Defrag function has several components that work together to optimize online reorganizations based on current database information. Figure 22 shows the online reorganization system flow.

Figure 22  Online reorganization system flow

The online reorganization includes the components that Table 18 on page 154 describes.
Table 18  Components of the online reorganization

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database</td>
<td>The HDAM or HIDAM database to be reorganized.</td>
</tr>
<tr>
<td>DBCTL</td>
<td>The database control feature that processes calls from application programs.</td>
</tr>
<tr>
<td></td>
<td>DBCTL communicates with the Online/Defrag job. The calls specify the reorganization actions to perform against the database records.</td>
</tr>
<tr>
<td>Online/Defrag job</td>
<td>The job scheduled to perform the online reorganization.</td>
</tr>
<tr>
<td></td>
<td>The Online/Defrag job performs the following tasks:</td>
</tr>
<tr>
<td></td>
<td>■ receives input (keywords) to control the reorganization parameters</td>
</tr>
<tr>
<td></td>
<td>When MODE(RECORD)&lt;Command&gt; is specified, the Online/Defrag job receives a key list data set as input that recommends records for reorganization. The recommendation is based on the Online/Defrag Record Analysis.</td>
</tr>
<tr>
<td></td>
<td>■ determines the free space available for reorganization</td>
</tr>
<tr>
<td></td>
<td>■ schedules a batch message processing (BMP) program to coordinate the reorganization with DBCTL</td>
</tr>
<tr>
<td></td>
<td>■ analyzes each record to determine candidacy for reorganization</td>
</tr>
<tr>
<td></td>
<td>■ schedules the eligible records for reorganization</td>
</tr>
<tr>
<td></td>
<td>■ automatically generates reports such as the Messages Log and Options in Effect</td>
</tr>
<tr>
<td>Keywords</td>
<td>Keywords control online reorganization parameters.</td>
</tr>
<tr>
<td>Key list data set</td>
<td>The key list data set contains the root segment key of the record that is recommended for reorganization.</td>
</tr>
<tr>
<td>Reports</td>
<td>Online/Defrag automatically generates reports that summarize reorganization activities.</td>
</tr>
</tbody>
</table>

**JCL requirements**

An explanation of all JCL used to perform an online reorganization follows:

**EXEC**

Required. You must specify PGM=BMCOLR.
STEPLIB DD
Optional. Defines the library that contains the Online/Defrag modules. The STEPLIB concatenation must contain an MDALIB for Online/Defrag to access if the RECON DD statements are not in the JCL. Always concatenate the BMC load library before the IMS library.

DFSRESLB DD
Required. Defines the library containing the RESLIB.

IMS DD
Required. Defines the library, typically IMSVS.DBDLIB, containing the DBD referenced in the Online/Defrag control statements.

PLUSIN DD
Required. Defines the control statement data set that contains commands and keywords for Online/Defrag to use during a reorganization.

OLRKEYS DD
Required when you specify MODE(RECORD) and you do not specify the KEYS() keyword. The OLRKEYS DD statement describes the data set that contains the record keys which Online/Defrag Analyzer recommends for reorganization. The record length can be variable or fixed, but it must be large enough to hold the root key. The OLRKEYS data set can be created by the Online/Defrag analysis.

RECONx DD
Required if no MDALIB is defined. The data sets named in the RECONx DD statements define the DBRC RECON data sets. BMC recommends that you do not include RECON DD statements in the JCL. Instead, add an MDALIB that contains the RECON data sets to your STEPLIB concatenations. RECON MDAs can be stored in the RESLIB.

BMCPRINT DD
Optional. Defines the output data set that contains reports. If the BMCPRINT DD statement is not included in the JCL, Online/Defrag dynamically allocates a BMCPRINT data set.

BMCMSG DD
Optional. Defines the output data set that contains messages and a listing of PLUSIN control statements. If the BMCMSG DD statement is not included in the JCL, Online/Defrag dynamically allocates a BMCMSG data set.
BMCTRACE DD

Optional. Defines the BMCTRACE DD statement to record diagnostic information about the job.

PDX DD

Optional. Defines the PDX data set name. Overrides the PDX data set name specified in the global options module. If the PDX data set name is not specified in the PDX DD statement, or the PDXDSN option in the global options module, PDX processing is bypassed. The internal default is NULLFILE.

Examples

Figure 23 shows the minimum JCL requirements for the online reorganization function. For DD statement descriptions, see “JCL requirements” on page 154.

Example JCL for Recording.

Figure 23  Minimum JCL requirements for Online/Defrag function

<table>
<thead>
<tr>
<th>DD Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>//BMCOLR</td>
<td>EXEC PGM=BMCOLR</td>
</tr>
<tr>
<td>//STEPLIB</td>
<td>DD DISP=SHR,DSN=load.dataSet</td>
</tr>
<tr>
<td>//DFSRESLB</td>
<td>DD DISP=SHR,DSN=dfs.dataSet</td>
</tr>
<tr>
<td>//IMS</td>
<td>DD DISP=SHR,DSN=ims.dataSet</td>
</tr>
<tr>
<td>//PLUSIN</td>
<td>DD *</td>
</tr>
<tr>
<td></td>
<td>(user options)</td>
</tr>
</tbody>
</table>

The primary difference among online reorganizations is the MODE specification. MODE is a keyword specified in the PLUSIN control statement. Three modes are available: record, scan, and block. For more information about each mode, see the following sections:

- “Record mode process” on page 137
- “Scan mode process” on page 139
- “Block Mode Process” on page 139

Record mode JCL

Figure 24 on page 157 shows sample JCL for a reorganization when MODE(RECORD) is specified. For a complete explanation of the Online/Defrag Record Analysis, see “Online/Defrag Record Analysis” on page 144.
For DD statement descriptions, see “JCL requirements” on page 154. The PLUSIN control statement specifies the reorganization parameters:

```
//PLUSIN   DD *

OREORG  DBD(DATABASE) DYNALLOC(Y) MODE(RECORD) -
         KEYS(DBRECORDKEY1, DBRECORDKEY2)
         BIPF(90) FSPF(100)
```

Dynamic allocation is specified, which requires that the primary database be registered in the RECON.

MODE(RECORD) requires an OLRKEYS DD statement or the KEYS keyword. The OLRKEYS DD statement defines the data set produced by the Online/Defrag Record Analysis. The KEYS keyword defines a hard-coded list of record keys. The record keys defined in the KEYS keyword identify records to be reorganized. If KEYS and OLRKEYS are specified, Online/Defrag processes the records that are specified in KEYS, then the records that are specified in OLRKEYS.

The BIPF keyword value is 90. The analysis identifies only those records that can be improved by 90 percent. For example, a record that spans 100 blocks must be capable of being reduced to 10 blocks. If the record cannot meet the specified percentage, the record is not recommended for reorganization.

The MBI keyword was not specified; therefore, the default value (5) is used. If a record cannot be reduced by at least 5 blocks, it is not be recommended for reorganization.

The FSPF keyword value is 100. Online/Defrag determines database blocks as available or not available, based on the FSPF value. If a block is not at least 100 percent free, it is not identified as available.
Scan mode JCL

Figure 25 shows sample JCL for a reorganization when MODE(SCAN) is specified.

For DD statement descriptions, see “JCL requirements” on page 154. The PLUSIN control statement specifies the reorganization parameters:

No keywords were specified to define the reorganization parameters. When no keyword values are specified, the default values are used: BIPF=25, MBI=5, FSPF=95. The Online/Defrag job scans the entire database and reorganizes every record that meets the criteria for reorganization.

The BIPF keyword value is 25. The analysis identifies only those records that can be improved by 25 percent. For example, a record that spans 100 blocks must be capable of being reduced to 75 blocks. If the record cannot meet the specified percentage, the record is not reorganized.

The MBI keyword was not specified; therefore, the default value (5) is used. If a record cannot be reduced by at least five blocks, it is not be reorganized.

The FSPF keyword value is 95. Online/Defrag determines database blocks as available or not available based on the FSPF value. If a block is not at least 95 percent free, it is not identified as available.

Note that dynamic allocation is requested. Dynamic allocation requires that the database be registered in the RECON data set. In Figure 25, the RECON data set names are specified in the MDALIB. If no MDALIB is specified, RECONx DD statements are required.
### Block mode JCL

Figure 26 shows sample JCL for a reorganization when MODE(BLOCK) is specified.

#### Figure 26 JCL for Online Reorganization block mode

```plaintext
//**************************************************************
//*     ONLINE REORG JCL FOR A DATABASE USING MODE(BLOCK)
//**************************************************************

//BMCOLR EXEC PGM=BMCOLR
//STEPLIB DD DISP=SHR,DSN=load.dataSet
//DFSRESLB DD DISP=SHR,DSN=dfs.stepLib
//IMS DD DISP=SHR,DSN=ims.dataSet
//DATABASE DD DISP=SHR,DSN=IMSVS.HIDAM.DATABASE
//RECON1 DD DISP=SHR,DSN=recon1.dataSet
//RECON2 DD DISP=SHR,DSN=recon2.dataSet
//RECON3 DD DISP=SHR,DSN=recon3.dataSet
//BMCMSG DD SYSOUT=*  
//BMCPRINT DD SYSOUT=*  
//BMCTRACE DD SYSOUT=*  
//PLUSIN DD *

OREORG MONITOR(300) DBD(DATABASE) PSB(PSBNAME) +  
SSID(IMSID) DYNALLOC(Y) FSPF(75) BIPF(80) MBI(1) +  
MODE(BLOCK)

//--
```

For DD statement descriptions, see “JCL requirements” on page 154. The PLUSIN control statement specifies the reorganization parameters:

OREORG DBD(DATABASE) MODE(BLOCK) SSID(IMSA)

When MODE(BLOCK) is specified, the entire database is reorganized.

### Options

For information about the keywords to use to customize database reorganizations, see Table 13 on page 141.

### Reports

Online/Defrag automatically generates reports. The statistics in the reports can help you make plans and decisions about future changes to your database.
Online/Defrag sends the reports to a certain destination as follows:

1. If you specified the OLRPRINT keyword, the reports are written to the OLRPRINT data set.

2. If you did not specify the OLRPRINT keyword but you specified the BMCPRINT DD statement, the reports are written to the BMCPRINT data set.

3. If you did not specify the OLRPRINT keyword or the BMCPRINT DD statement, Online/Defrag dynamically allocates the BMCPRINT data set.

Table 19 lists and describes reports generated by Online/Defrag. Refer to the corresponding page number for more information about each report.

<table>
<thead>
<tr>
<th>Report</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Log</td>
<td>Lists the options specified in the PLUSIN control statement and the messages issued during the job execution.</td>
<td>160</td>
</tr>
<tr>
<td>Options in Effect</td>
<td>Lists the options in effect for the DBD.</td>
<td>160</td>
</tr>
<tr>
<td>DBD Summary</td>
<td>Lists DBD specifications at the time of reorganization.</td>
<td>162</td>
</tr>
<tr>
<td>Reorganization Results</td>
<td>Lists results of the online reorganization.</td>
<td>163</td>
</tr>
</tbody>
</table>

**Banner page**

All reports are preceded by a banner page.

**Message log**

The message log lists the options specified in the PLUSIN control statement and the messages that Online/Defrag issues during the job execution. Messages are generated automatically and written to the BMCMSG data set.

**Options in Effect**

The Options in Effect report lists the options in effect for the DBD. The report provides the name of the option, the value, and the source of the value. Values are internal defaults or values that are specified in the PLUSIN control statement. The report is generated automatically and written to the BMCPRINT data set.
The Options in Effect report provides the following information:

### Table 20  Options in Effect report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS APPLICATION GROUP NAME</td>
<td>AGN</td>
<td>The name of the application group. The AGN works with your security product to provide security for the Online/Defrag job. The value is blank if AGN was not specified.</td>
</tr>
<tr>
<td>BLOCK IMPROVEMENT % FACTOR</td>
<td>BIPF</td>
<td>The percentage of block improvement to qualify the reorganization.</td>
</tr>
<tr>
<td>DBD NAME</td>
<td>DBD</td>
<td>The name of the description for the primary database.</td>
</tr>
<tr>
<td>WTO DESCRIPTOR CODES</td>
<td>DESCODE</td>
<td>The number of descriptor codes used to route messages to the console. Descriptor codes describe the significance of the messages.</td>
</tr>
<tr>
<td>USE DYNAMIC ALLOCATION</td>
<td>DYNALLOC</td>
<td>The dynamic allocation specification.</td>
</tr>
<tr>
<td>BLOCK FREE SPACE % FACTOR</td>
<td>FSPF</td>
<td>The percentage of free space to indicate that a block is free.</td>
</tr>
<tr>
<td>MINIMUM BLOCK IMPROVEMENT</td>
<td>MBI</td>
<td>The minimum block improvement.</td>
</tr>
<tr>
<td>REORG MODE</td>
<td>MODE</td>
<td>The reorganization mode. Values: RECORD, SCAN, and BLOCK.</td>
</tr>
<tr>
<td>MONITOR REORG PROGRESS</td>
<td>MONITOR</td>
<td>The status of monitor access during an online reorganization. Values: Y and N.</td>
</tr>
<tr>
<td>INTERVAL BETWEEN MESSAGES</td>
<td>MONITOR</td>
<td>The time interval value assigned to the MONITOR keyword. Online/Defrag issues a message at the specified interval.</td>
</tr>
<tr>
<td>REPORT DD</td>
<td>OLRPRINT</td>
<td>The data set that contains reports which were generated. The default data set is BMCPRINT.</td>
</tr>
<tr>
<td>WTO ROUTE CODES</td>
<td>ROUTCODE</td>
<td>The number of routing codes used to control destination of messages sent to the console. Default values: 2 (master console) and 7 (unit record pool).</td>
</tr>
<tr>
<td>IMS ID OF ONLINE CTL REGION</td>
<td>SSID</td>
<td>The IMSID used to connect to the IMS control region.</td>
</tr>
<tr>
<td>IMS RELEASE LEVEL</td>
<td>N/A</td>
<td>The current IMS release level. The level does not indicate system requirements for Online/Defrag.</td>
</tr>
<tr>
<td>STEPLIB CONCATENATION</td>
<td>N/A</td>
<td>The order in which the STEPLIB data sets are concatenated.</td>
</tr>
<tr>
<td>DBD LIBRARY NAME</td>
<td>N/A</td>
<td>The name of the DBD library.</td>
</tr>
<tr>
<td>DBRC DATA SETS</td>
<td>N/A</td>
<td>The names of the RECON data sets.</td>
</tr>
<tr>
<td>KEY LIST DATA SET</td>
<td>N/A</td>
<td>The name of the key list data set.</td>
</tr>
<tr>
<td>DATA BASE DATA SETS</td>
<td>N/A</td>
<td>The name of the database data sets.</td>
</tr>
</tbody>
</table>
### DBD Summary

The DBD Summary report shows the status and the specification in the DBD at the time of reorganization. The summary is generated automatically.

The DBD Summary report provides the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE</td>
<td>The name of the primary database.</td>
</tr>
<tr>
<td>DS GROUPS</td>
<td>The number of data set groups.</td>
</tr>
<tr>
<td>ORGANIZATION</td>
<td>The database organization.</td>
</tr>
<tr>
<td>ACCESS METHOD</td>
<td>The access method used.</td>
</tr>
<tr>
<td>RANDOMIZER</td>
<td>The name of the HDAM randomizer used.</td>
</tr>
<tr>
<td>NUMBER RAPS</td>
<td>The number of root anchor points (RAPs) in the HDAM parameters.</td>
</tr>
<tr>
<td>MAX RBN</td>
<td>The maximum RBNs in the RAA that were specified in the HDAM parameters.</td>
</tr>
<tr>
<td>MAX BYTES</td>
<td>The maximum number of bytes specified in the HDAM parameters that will be written to the RAA.</td>
</tr>
<tr>
<td>SEGMENT</td>
<td>The segment name.</td>
</tr>
<tr>
<td>SEG CODE</td>
<td>A one-byte hexadecimal value that identifies the segment in the database.</td>
</tr>
<tr>
<td>LVL</td>
<td>The hierarchical level of the segment.</td>
</tr>
<tr>
<td>DSG</td>
<td>The data set group of the segment.</td>
</tr>
<tr>
<td>PARENT</td>
<td>The name of the segment parent.</td>
</tr>
<tr>
<td>PFX LEN</td>
<td>The prefix length, in bytes, of the segment.</td>
</tr>
<tr>
<td>DATA LEN</td>
<td>The data length, in bytes, of the segment. If the segment has a variable length, this field displays the maximum data length.</td>
</tr>
<tr>
<td>FLAG</td>
<td>This field can contain one of the following values:</td>
</tr>
<tr>
<td></td>
<td>- P—The segment was padded with one byte to make it an even length in the database. IMS does not write odd-length segments in HD-type databases. You could increase the segment length in the DBD by one byte and not use any more disk space.</td>
</tr>
<tr>
<td></td>
<td>- V—The segment is defined as variable length.</td>
</tr>
<tr>
<td></td>
<td>- C—The segment has an edit/compression routine.</td>
</tr>
<tr>
<td>PTR TYPE</td>
<td>The types of pointers contained in the segment prefix, listed in the same order as they appear in the segment prefix. If hierarchical pointers are used, the target segment type may vary depending on the other segments in the hierarchical chain.</td>
</tr>
</tbody>
</table>
Reorganization Results

The Reorganization Results report shows the results of an online reorganization. The report is automatically generated when block mode or scan mode is specified. Reorganization results for record mode are presented for each record in the messages output.

The Reorganization Results report provides the following information:

Table 22  Reorganization Results report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORD COUNT</td>
<td>The number of reorganized records that achieved the same reorganization results.</td>
</tr>
<tr>
<td>OLD BLOCKS</td>
<td>The number of blocks that the specified records spanned prior to the reorganization.</td>
</tr>
<tr>
<td>NEW BLOCKS</td>
<td>The number of blocks that the specified records span after the reorganization.</td>
</tr>
<tr>
<td>% IMPROVEMENT</td>
<td>The percentage that the record was improved. For example, the percentage improvement is 50 percent if the record spans half the number of blocks that it spanned prior to the reorganization.</td>
</tr>
</tbody>
</table>
Multiple databases

This chapter describes how to reorganize and copy multiple databases during the same job step. The chapter includes the following topics:

Overview ................................................................. 165
Requirements ............................................................ 167
Commands and keywords ............................................. 168
JCL ................................................................. 169
Reports ............................................................... 170

Overview

BMC introduces support for copying and reorganizing multiple databases simultaneously. The benefits of copying and reorganizing multiple databases in one job step include:

- reorganizing and copying all logically related databases
- reorganizing and copying all databases in an application group
- reducing log switches

Figure 27 on page 166 shows the task flow of an online reorganization job for a multiple database reorganization. Refer to the text explanation that follows.
The following steps describe the multiple database reorganization process for the Online/Reorg function:

1. **Preprocessing**
   
   `/BMCPAU` is issued to quiesce the databases. Log recording is initiated. `/STA` is issued to start the databases and release any suspended transactions.

2. **Copy**

   Each database is assigned to a separate unit of work (UOW) and copied.

3. **Post copy**

   `/BMCPAU` is issued to quiesce the databases. Log recording is terminated. Logs are applied. Change recording is initiated. `/STA` is issued to start the databases and release any suspended transactions.

4. **Reorg**

   Each database is assigned to a separate UOW and reorganized.

5. **Post processing**

   Prefix resolution and updates are performed. `/DBR` is issued to stop the databases. Changes are applied. Change recording is terminated. `/STA` is issued to start databases. Data sets are swapped. DBRC functions are performed. You can run only one restart job. You can include the name of any one of the databases from the GROUP subcommand for the PARM keyword, but only one restart job should be run. Restart automatically runs for all of the databases.
Cross-System Image Manager (XIM) is required to copy or reorganize multiple database simultaneously. For information about XIM, refer to Appendix G, “Cross-System Image Manager.”

The following requirements apply to multiple database reorganizations:

- XIM must be installed and active.
- You must have sufficient XIM initiators. The number of initiators must be equal to or greater than the maximum number of databases being reorganized. You can verify or specify the number of initiators in the XIM control file.
- You must remove the database name from the PARM statement on the EXEC command. Use the GROUP subcommand to specify database names.
- For the Online Reorg function, all logically related databases must be reorganized simultaneously.

You can have one DBD per UOW, but not one data set per UOW. The only way to REORG multiple databases is to use the GROUP keyword. It is possible to code just one DBD on a single GROUP. In this case, you do not use any UOWs, the database would REORG in the home address space. However, anytime you code multiple DBDs on GROUP statement(s), each database is reorganized in a separate UOW. When reorganizing in UOWs, any of the databases in the home address space are not reorganized. The home address space is used to validate control card information, start and monitor the UOWs, and once the complete to do PRP and post processing. If it is a CRF job with multiple databases, a separate UOW is used to copy each of the databases. In a stand-alone COPY job, if you specify multiple COPY control statements, a UOW is used to copy each of the databases.

The following requirements apply to multiple database copies:

- XIM must be installed and active.
- You must have sufficient XIM initiators. The number of initiators must be equal to or greater than the maximum number of databases being reorganized. You can verify or specify the number of initiators in the XIM control file.
- You must specify multiple COPY statements.
Table 23 shows the commands that enable multiple database reorganizations. For all Reorg function keywords, refer to Chapter 3, “Online Reorg function”; for all Online Reorg function keywords, refer to Chapter 2, “Reorg function (Fast Reorg Facility/EP).” For more information about the Copy function, refer to Chapter 9, “Copy function.”

Table 23  Commands for multiple database reorganizations and COPY

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online Reorg</td>
<td>REORG</td>
<td>The REORG command serves as a global statement. It supplies job-level</td>
</tr>
<tr>
<td>Reorg</td>
<td>GROUP</td>
<td>BMC introduces the GROUP subcommand. It is specified with the REORG</td>
</tr>
<tr>
<td></td>
<td></td>
<td>command and provides DBD-level definitions. Parameters specified in the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GROUP statement override parameters specified in the REORG statement.</td>
</tr>
<tr>
<td>Copy</td>
<td>COPY</td>
<td>Multiple COPY statements are allowed so that you can copy multiple</td>
</tr>
<tr>
<td></td>
<td></td>
<td>databases during one job step.</td>
</tr>
</tbody>
</table>

Table 24 shows keyword changes for multiple database reorganizations and copies.

Table 24  Keyword changes for multiple database reorganizations and copies

<table>
<thead>
<tr>
<th>Command</th>
<th>Keyword</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG</td>
<td>DBRC</td>
<td>Support was added for DBRC.</td>
</tr>
<tr>
<td>GROUP</td>
<td>DBD</td>
<td>Required. You must remove the DBD keyword from the EXEC statement.</td>
</tr>
<tr>
<td>COPY</td>
<td>DYNALLOC</td>
<td>DYNALLOC is mutually exclusive with INDD/OUTDD.</td>
</tr>
<tr>
<td>IDCAMS</td>
<td></td>
<td>Support was added for IDCAMS.</td>
</tr>
<tr>
<td>INDD and OUTDD</td>
<td></td>
<td>If you specify INDD and OUTDD, a single COPY statement can only copy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>one data set; therefore, you must have a COPY statement for each data set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDD/OUTDD is mutually exclusive with DYNALLOC.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INDD/OUTDD cannot specify the ILDS of a HALDB. (If DYNALLOC is used for HALDB, the ILDS is automatically copied.)</td>
</tr>
<tr>
<td>SECINDEX</td>
<td></td>
<td>The SECINDEX keyword on the COPY statement is valid but requires dynamic allocation.</td>
</tr>
</tbody>
</table>
The JCL requirements to run multiple database reorganizations and copies are equivalent to single database reorganizations. If you are using the Online Reorg function, refer to Chapter 3, “Online Reorg function”; if you are using the Reorg function, refer to Chapter 2, “Reorg function (Fast Reorg Facility/EP).” BMC strongly recommends that you use dynamic allocation.

**NOTE**

When using the Online Reorg function, do not include the database DDs in any step after the reorganization step. If you do, the online region might get a dynamic allocation error when the Online Reorg function issues the /STA command after the swap process.

Figure 28 and Figure 29 show sample PLUSIN statements.

### Figure 28  REORG PLUSIN example for multiple databases

```jcl
//PLUSIN DD *
   REORG DYNALLOC(Y,Y,Q) DBRC(Y)
   GROUP DBD(DB01,DB02) ICP(Y)
   GROUP DBD(DB03,DB04)
```

The JCL control cards in Figure 28 execute a multiple database reorganization of databases DB01, DB02, DB03, and DB04 with DBRC active for all reorganizations. The Image Copy function only runs for DB01 and DB02.

### Figure 29  COPY PLUSIN example for multiple databases

```jcl
/*
//PLUSIN DD *
COPY +
   DBRC(Y) +
   DBD(DB01) IMSID(CRFH) SWAP(N) DYNALLOC(YES,YES,Z)
COPY +
   DBD(DB02) IMSID(CRFH) SWAP(N) DYNALLOC(YES,YES,Z)
/*
/*
```

The JCL control cards in Figure 29 execute a multiple database copy of databases DB01 and DB02. The monitoring IMS system is CRFH. The databases are not swapped after the copy. Dynamic allocation is used, and the copied file has a .Z suffix.
Figure 30 shows sample JCL for a multiple database reorganization.

**Figure 30  Reorg function JCL for multiple database reorganization**

```plaintext
//STEP1 EXEC PGM=DFSRRC00,PARM='ULU,DFSURPRO'
//STEPLIB DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//SYSPRINT DD SYSOUT=*  //DFSURCDS DD DISP=(,CATLG),DSN=dfsurcds,UNIT=SYSDA,
//SYSPRINT DD SYSOUT=* DCB=BLKSIZE=1600,SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=*
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSURCDS DD DISP=OLD,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//PLUSIN DD *

//ALLOC EXEC PGM=DFSRRC00,PARM='ULU,DFSURPRO'
//STEPLIB DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//SYSPRINT DD SYSOUT=*  //DFSURCDS DD DISP=(,CATLG),DSN=dfsurcds,UNIT=SYSDA,
//SYSPRINT DD SYSOUT=* DCB=BLKSIZE=1600,SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=*  
```

**Reports**

Reports for a multiple database reorganization are generated for each database as processing completes. Figure 31 on page 171 shows sample output from a multiple database reorganization.
UOWs are identified by the ddname suffix. The final characters of the ddname are overlaid with the UOW ID. For example, BMCMSG01 contains the messages for UOW1. BMCMSG is the primary message output data set and contains a road map of UOW assignments. For a sample BMCMSG report, refer to Figure 32 on page 172.

The output shown in Figure 31 indicates that four databases were reorganized simultaneously. The index lists four BMCMSGx, BMCTRACx, and BMCPRNx data sets. In addition, the index lists two Prefix Resolution and Update sort reports and one Image Copy report.

Figure 32 on page 172 is an excerpt from a sample BMCMSG report for a multiple database copy. This output shows assignment of four UOWs. One UOW is assigned to each database copy task. The BMCMSG output displays status of the UOWs in each task. Assignment of UOWs is the same for multiple database reorganizations and copies.
Abends

If a UOW abends, error output is generated from each UOW address space. BMCMSG reports the error messages and identifies the UOW by displaying the UOW ID between the message number and text. The JES log and JES messages are also generated for each UOW that abends and can be identified by the UOW ID. The following example shows the format of error output in BMCMSG.

Options in Effect

The Options in Effect Report has a new option for multiple database reorganizations. The new option in effect is XIMGROUP=DBUXIM00. This option is specified in the setup macro and identifies the XIM group name as specified in the XIM parameter member.
Unload function

This chapter describes how to run the Unload function as a stand-alone utility. This chapter includes the following topics:

Overview ............................................................. 174
JCL requirements ....................................................... 175
   Non-parallel unload .................................................. 175
   Parallel unload ....................................................... 176
DD statements .......................................................... 176
PLUSIN examples ....................................................... 177
Snapshot Copy function ................................................ 178
Keywords ............................................................... 178
Global options modules ............................................... 181
   Create using ISPF interface ......................................... 182
   Create using batch job ............................................... 184
Reports ................................................................. 184
HD Unload data set differences ....................................... 185
Performance considerations ........................................... 186
   Buffers ............................................................... 186
   Options ............................................................. 187
DB Scan function ....................................................... 187
   JCL requirements ................................................... 189
Database Selection facility ............................................. 192
   JCL requirements ................................................... 192
   Examples ........................................................... 195
   Output ............................................................... 196
API ..................................................................... 199
   System flow ........................................................ 200
   JCL requirements ................................................... 202
   USERREAD command .............................................. 204
   TRACE command .................................................. 205
   UNLOAD command ................................................ 206
   Restrictions ......................................................... 207
   HSSR conversion utility ............................................ 209
   Reports ............................................................. 210
The Unload function (invoked with the UNLOAD command) replaces the IMS HD Reorganization Unload utility. Figure 33 shows the system flow of the Unload function.

When you execute the UNLOAD command, the first program to be executed is the BMC version of the batch region controller (DFSRRC00), which loads and transfers control to the Unload function.

When the Unload function receives control, it determines the values to use for the user-controlled options as explained in "How the utilities build options" on page 385. The Unload function builds control blocks, opens data sets, and begins to unload the database.

The Unload function retrieves segments from the database and places them into the HD Unload data set. The Unload function uses its own retrieval methods rather than DL/I to search the database.
As the Unload function unloads the database, it accumulates statistics (for example, segment placement information and database record sizes). After the unload, the Unload function closes the data sets and generates reports, which can be stored in the PDX. These reports, and previous reports for the same database, can be viewed at any time through the ISPF interface.

---

**NOTE**

Reports pertaining to HALDBs and the IMS/ESA Partition Support Product with a partitioned primary index cannot be stored in the PDX. Report information for HALDBs is in the SYSOUT statement.

If DBRC is active, the Unload function works with it in the same manner as the IMS HD Reorganization Unload utility.

### JCL requirements

The JCL requirements for the HD Unload function are similar to those for the IMS HD Reorganization Unload utility.

### Non-parallel unload

Figure 34 shows the JCL required for a non-parallel unload for database with multiple data set groups.

**Figure 34** Unload function JCL (non-parallel mode)

```
//UNLOAD   EXEC PGM=DFSRRC00,PARM='ULU,DFSURGOU,dbdName'
//UNLOAD   EXEC PGM=DFSRRC00,PARM='ULU,DFSURGOU,dbdName'
//STEPLIB  DD DISP=SHR,DSN=bmc.xxx.load
  //  DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS  DD DISP=SHR,DSN=IMSVS.DBDLIB
//RECON1  DD DISP=SHR,DSN=your.recon1  ===>for HALDB only; required
//RECON2  DD DISP=SHR,DSN=your.recon2  ===>for HALDB only; required
//RECON3  DD DISP=SHR,DSN=your.recon3  ===>for HALDB only; required
//DFSURGOU1 DD DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(1,1),RLSE),
  DSN=test.hdunld
//dbddsg1 DD DISP=OLD,DSN=dataSetName.dsg1
//dbddsg2 DD DISP=OLD,DSN=dataSetName.dsg2
//dbddsg3 DD DISP=OLD,DSN=dataSetName.dsg3
//BMCMSG  DD SYSOUT=*  
//BMCPRINT DD SYSOUT=*  
//PLUSIN  DD *  
UNLOAD
```
Parallel unload

Figure 35 shows the JCL requirements for a parallel unload.

**Figure 35  Unload function JCL (parallel mode)**

```plaintext
//UNLOAD   EXEC PGM=DFSRRC00,PARM='ULU,DFSURGU0,dbName'
//STEPLIB  DD DISP=SHR,DSN=bmc.xxx.load
//   DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLIB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS     DD DISP=SHR,DSN=IMSVS.DBDLIB
//RECON1 DD DISP=SHR,DSN=your.recon1 //===for HALDB only: required
//RECON2 DD DISP=SHR,DSN=your.recon2 //===for HALDB only: required
//RECON3 DD DISP=SHR,DSN=your.recon3 //===for HALDB only: required
//DFSURGU1 DD DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(1,1),RLSE),
   DSN=test.hdunld1
//DFSURGU2 DD DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(1,1),RLSE),
   DSN=test.hdunld2
//DFSURGU3 DD DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(1,1),RLSE),
   DSN=test.hdunld3
//dbds    DD DISP=OLD,DSN=database.dataSetName
//BMCMSG  DD SYSOUT=*  //===for HALDB only: required
//BMCPRINT DD SYSOUT=*  //===for HALDB only: required
//PLUSIN  DD *
//UNLOAD   LIMITS(1,500,DFSURGU1,501,1000,DFSURGU2,1001,1500,DFSURGU3)
```

**DD statements**

For more information about the DD statements, see Chapter 13, “JCL requirements.”

The following DD statements are specific to the Unload function:

**dbddsgn/dbds DD**

Optional; however, if you specify DYNALLOC(YES) and the data sets are registered to DBRC, the Unload function dynamically allocates these data sets. Do not include this DD statement if you specify DYNALLOC(YES). Defines the data sets that contain the database being unloaded. You must supply one DD statement for each data set group in the DBD. The ddnames must match the ddnames in the DBD or be specified with the HALDB Partition Definition Utility (PDU).

You might need to supply DD statements for the data sets in any databases logically related to the database you are unloading. If the database you are unloading contains any logical child segments with virtual logical parent concatenated keys (LPCKs), you must supply the DD statements. If all LPCKs are physical, you do not need to supply DD statements for the data sets containing the logical parent segments.
newdbd DD

Required if you specify the NEWDBD keyword. Describes the data set that contains the DBD for which you are changing characteristics by using the NEWDBD keyword.

ULPSNAP DD

Optional. The Unload function dynamically allocates this data set if it is needed. Describes the data set that contains data blocks with pointer and/or sequence errors.

PLUSIN examples

The following examples show some possible combinations of keywords:

Example 1:
This example illustrates how to create the most abbreviated unload file. The database is unloaded in extra short format, and the compressed segments are not expanded:

```
UNLOAD DFSURGUI(XSHORT) EXPAND(NO)
```

Example 2:
The unload database data sets are dynamically allocated. The data sets must be registered to DBRC:

```
UNLOAD DYNALLOC(YES)
```

Example 3:
Structural changes are made to the database according to the DBD in the DBDLIB associated with the data set with the ddname DBDLIST:

```
UNLOAD NEWDBD(DBDLIST)
```

Example 4:
The unload reports are written to the data set with the ddname ULPRPTS. A corresponding DD statement can be included in the JCL:

```
UNLOAD ULPPRINT(ULPRPTS)
```
Snapshot Copy function

The Snapshot Copy function requires that you have a licensed copy of XBM for IMS version 4.4 or later or SUF for IMS version 4.2 or later. For more information, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide and the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE Installation Guide manuals.

If the database is on a qualified intelligent storage device, the Snapshot Copy function uses the device hardware to read the pre-updated database blocks. If qualified intelligent storage devices are not available, the Snapshot Copy function uses the XBM cache. In this case, the XBM subsystem intercepts any updates made to an XBM-registered database and preserves the pre-images in cache. (Pre-images are copies of database blocks before they are updated; cache is virtual storage obtained by the XBM subsystem.) The Unload function includes the pre-images in the Snapshot Copy instead of the corresponding blocks read from the database.

Snapshot Copy requires DBRC. All databases and data sets must be registered to DBRC.

Snapshot Copy must be invoked when the database is set up for reorganization. To ensure that no changes are made, the Unload function uses the automated operator interface (AOI) to issue /DBDs for the required database. If necessary, the Unload function changes the DBRC authorization to prevent updates by any batch jobs while Snapshot Copy is being started for the database data sets. After Snapshot Copy is started, a /STA command is issued and DBRC authorization is returned to the original state.

To protect the database from updates while it is being set up, the RECONs used must be shared with all applications having update capability for the database. To allow the AOI to issue commands where needed, the BMC load library must be included in all IMS control regions and DBCTL regions that share RECON data sets, as well as the DBRC region.

To protect the database data sets from updates in a sysplex environment during the unload, specify an IRLM group name. For more information, refer to “IRLMGRP” on page 477.

Keywords

The Unload function uses the DBSCAN and UNLOAD commands. Table 25 on page 179 lists valid keywords.
### Table 25  Unload function keywords (part 1 of 3)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFSPACE</td>
<td>(1-999)</td>
<td>(20)</td>
<td>413</td>
<td>Allows a buffer space specification of up to 999 MB for the unload step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>While these buffers are not page fixed, they can have a significant impact</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>on your system resources and should be used with caution.</td>
</tr>
<tr>
<td>CREORG</td>
<td>(YES,OKAYnnnn)</td>
<td>(YES,OKAYnnnn)</td>
<td>419</td>
<td>Specify conditional reorganization option for MAXM Database Advisor for IMS.</td>
</tr>
<tr>
<td></td>
<td>(YES,STOPnnnn)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DESCCODE</td>
<td>(mn,...)</td>
<td>(7)</td>
<td>490</td>
<td>Specify descriptor codes used to route messages to the console.</td>
</tr>
<tr>
<td>HSAMOUT</td>
<td>(dbname)</td>
<td>blank</td>
<td>448</td>
<td>Write an HSAM database.</td>
</tr>
<tr>
<td>IMSID</td>
<td>(imsID)</td>
<td>Use the IMSID</td>
<td>470</td>
<td>Specify IMSID under which reports are stored in the PDX.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>from IMS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>generation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DFSVC000 module).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRLMGRP</td>
<td>(irlmgrp)</td>
<td>none</td>
<td>477</td>
<td>Specify the IRLM group to be used with the Snapshot Copy function.</td>
</tr>
<tr>
<td>LIMITS</td>
<td>(startVal, endVal,</td>
<td>none</td>
<td>479</td>
<td>Specify subsets for a parallel unload.</td>
</tr>
<tr>
<td></td>
<td>ddname,...)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(AUTO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(PART)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPCK</td>
<td>(NO)</td>
<td>(YES)</td>
<td>487</td>
<td>Retrieve the logical parent concatenated key (LPCK).</td>
</tr>
<tr>
<td>MONITOR</td>
<td>(nnnnnnnn)</td>
<td>no automatic</td>
<td>490</td>
<td>Monitor job progress.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>monitoring</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Keywords

#### Table 25  Unload function keywords (part 2 of 3)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MONUSERS</td>
<td>(user01, user02,...)</td>
<td>Send monitor messages to TSO users and BMCMSG.</td>
<td>490</td>
<td>Send monitor messages to up to 10 specified TSO user IDs.</td>
</tr>
<tr>
<td>NEWDBD</td>
<td>(ddname)</td>
<td>none</td>
<td>494</td>
<td>Restructure the database using this new DBD.</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>(YES) (NO)</td>
<td>(YES)</td>
<td>504</td>
<td>Enable parallel processing.</td>
</tr>
<tr>
<td>PART</td>
<td>(ALL) (partitionName,...) (nn,nn,...) (ALL)</td>
<td>Perform reorganization tasks for one or more database partitions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDXPARMS</td>
<td>(userName)</td>
<td>Use the DBD name specified in the EXEC statement’s PARM operand.</td>
<td>520</td>
<td>Override the DBDname member in the PDX.</td>
</tr>
<tr>
<td>PTRCHECK</td>
<td>(NO) (PHY)</td>
<td>(NO)</td>
<td>524</td>
<td>Verify pointers.</td>
</tr>
<tr>
<td>PTRERROR</td>
<td>(ABEND) (ACCEPT) (ACCEPT, n) (ABEND)</td>
<td>Specify the action to take if a pointer error is found.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>REPORTS</td>
<td>See page 526</td>
<td>See page 526</td>
<td>526</td>
<td>Generate reports.</td>
</tr>
<tr>
<td>ROUTCODE</td>
<td>(nn,...) max 10 entries</td>
<td>(2,7)</td>
<td>Specify routing codes to control destination of messages sent to the console.</td>
<td></td>
</tr>
<tr>
<td>SEGSEL</td>
<td>(segName1, segName2,...) max 30 names</td>
<td>none</td>
<td>Unload only the specified segment names.</td>
<td></td>
</tr>
<tr>
<td>SEQERROR</td>
<td>(ABEND) (ACCEPT) (ACCEPT,n) (ABEND)</td>
<td>Specify the action to take if a sequence error is found.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SNAPSHOT</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>543</td>
<td>Specify whether to take a Snapshot Copy of the database being unloaded.</td>
</tr>
<tr>
<td>SYNCTAPE</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>554</td>
<td>Synchronize unload output tape volumes.</td>
</tr>
<tr>
<td>ULPPRINT</td>
<td>(ddname)</td>
<td>BMCPRINT</td>
<td>522</td>
<td>Specify the data set that will contain the Unload function reports and processing output.</td>
</tr>
<tr>
<td>UNLAUTH</td>
<td>(N) (Y)</td>
<td>(Y)</td>
<td>559</td>
<td>Specify whether to obtain authorization for an unload.</td>
</tr>
<tr>
<td>USEINDEX</td>
<td>(YES) (dbname) (NO)</td>
<td>(NO)</td>
<td>Unload roots through an index.</td>
<td></td>
</tr>
</tbody>
</table>
You can specify options for the Unload function at the levels shown in Table 26. Because the Unload function uses the UNLOAD PLUS global option modules, not all of the Unload function keywords are available in a global options module.

**NOTE**

You can override any option specified in an options module by including that option in the PLUSIN control statement data set.

<table>
<thead>
<tr>
<th>Module</th>
<th>Used for</th>
<th>Resides in</th>
<th>Created by</th>
</tr>
</thead>
<tbody>
<tr>
<td>internal defaults</td>
<td>all jobs that do not specify an options module or options in PLUSIN</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>ULPPARMS</td>
<td>all jobs at your installation</td>
<td>load library</td>
<td>ISPF panels or #DBUGLBL batch job</td>
</tr>
<tr>
<td>ULP@imsi</td>
<td>all jobs run on a particular IMS system (for example, a test system)</td>
<td>load library</td>
<td>ISPF panels or #DBUGLBL batch job</td>
</tr>
<tr>
<td>DFLT member of PDX</td>
<td>all jobs, all jobs run on a particular IMS system, or all jobs for a particular database, depending on how the options module was created</td>
<td>PDX</td>
<td>an earlier release of the product</td>
</tr>
<tr>
<td>dbdname member of PDX</td>
<td>all jobs for a particular database; PDX not supported for HALDBs</td>
<td>PDX member</td>
<td>ISPF panels</td>
</tr>
<tr>
<td>PLUSIN</td>
<td>job where the options are specified</td>
<td>JCL</td>
<td>adding keywords to PLUSIN control statement data set in reload job</td>
</tr>
</tbody>
</table>

Table 26 Ways to specify options for the Unload function

**Table 25 Unload function keywords (part 3 of 3)**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>USEREXIT</td>
<td>(exitName,, YES</td>
<td>none</td>
<td>566</td>
<td>Invoke a user exit routine.</td>
</tr>
<tr>
<td>USERHDR</td>
<td>See page 568</td>
<td>none</td>
<td>568</td>
<td>Create output records with a user-defined header (or prefix) to segment data.</td>
</tr>
<tr>
<td>XBMSSID</td>
<td>(ssid)</td>
<td>none</td>
<td>574</td>
<td>Specify the XBM subsystem to be used with the Snapshot Copy function.</td>
</tr>
</tbody>
</table>
You can combine the global options module, the PDX options member, and command methods for specifying options for maximum efficiency. You can use the ULPPARMS options module to specify the values you want to use for most of the databases at your site. For particular keywords you want to use only on a single IMS system (for example, a test system), you can use the ULP@imsi options module. For databases that need values different from those you specified in the ULPPARMS or ULP@imsi modules, you can create a PDX options member for those DBDs. For jobs that require values different from those you usually use for the database, you can specify the values in the PLUSIN data set.

You can specify values for the following options in a global options module for use with the Unload function:

- DFSURGU1
- EXPAND
- LPCK
- PDX
- PDXPARMS
- PTRCHECK
- PTRERROR
- SEQERROR
- SYNCTAPE
- USEREXIT

If you specify options that the Unload function does not use (for example, FASTIO), those options are ignored.

Create using ISPF interface

You can create and modify the ULPPARMS and ULP@imsi global options modules by using the Unload function ISPF interface. You can also create the DBD-level global members that are stored in the PDX. Figure 36 on page 183 shows the hierarchy of the interface.
The ISPF interface makes it easy for you to specify options at the IMSID and/or DBD level. Enter the required information on one of the global options panels. During execution, the utility retrieves settings from the global options module. Each IMSID and DBD can have a set of options associated with it.

Create and modify options as follows:

- To create a set of options for a particular DBD, select option 2 from the Unload Function primary menu.

- To create or modify a global options module (ULPPARMS or ULP@imsi), select option 3 from the Unload Function primary menu.

To scroll through the options panels, press Enter. When you press Enter, the utility validates the options you specified on the current panel and displays the next panel. The scroll keys (UP and DOWN) are not active. To go to a particular panel, enter its page number in the Next Page field of the current panel and press Enter.

When you make changes on these panels, you must press Enter before the utility validates your changes.
An explicit save is performed when you type **SAVE** on the **Command** line of a panel where **SAVE** is displayed as a valid command. If you do not want to save your changes, type **CANCEL** on the **Command** line and press **Enter**.

When you are finished, perform an implicit save by typing **END** on the **Command** line or pressing the function key designated as your **END** key (usually **F3**).

**NOTE**

Any keyword specified in the PLUSIN control statement data set overrides the corresponding keyword in the global options module.

Online Help is available for each panel in the ISPF interface.

## Create using batch job

You can create and modify the ULPPARMS and ULP@imsi global options modules by executing the two ULPGLBLx steps in the #DBGGLBL member of the sample library.

## Reports

The Unload function generates the reports shown in **Table 27**.

**Table 27  Unload function reports (part 1 of 2)**

<table>
<thead>
<tr>
<th>Report</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Selection Facility (DBSF) output (if you invoke the DBSF)</td>
<td>196</td>
</tr>
<tr>
<td>Dataset Statistics</td>
<td>591</td>
</tr>
<tr>
<td>DBD Summary</td>
<td>592</td>
</tr>
<tr>
<td>Disassembled DBD</td>
<td>594</td>
</tr>
<tr>
<td>Distribution of Database Record Sizes</td>
<td>594</td>
</tr>
<tr>
<td>Distribution of HDAM RAP Chain Lengths</td>
<td>595</td>
</tr>
<tr>
<td>HDAM Root Addressable Area Statistics</td>
<td>597</td>
</tr>
<tr>
<td>HDAM Root Placement Relative to Its Home Block</td>
<td>598</td>
</tr>
<tr>
<td>Largest Database Records</td>
<td>600</td>
</tr>
<tr>
<td>Logical Relationship Information</td>
<td>601</td>
</tr>
<tr>
<td>Message Log</td>
<td>604</td>
</tr>
<tr>
<td>Options in Effect</td>
<td>604</td>
</tr>
<tr>
<td>Parent/Child Segment Statistics</td>
<td>606</td>
</tr>
</tbody>
</table>
The HD Unload data set that is produced by the Unload function differs from the HD Unload data set produced by the IMS HD Reorganization Unload utility in the following ways:

- The HD Unload data set that is produced by the Unload function is not a byte-for-byte duplicate of the HD Unload data set that is produced by the IMS HD Reorganization Unload utility. The Unload function produces certain additional header records (X’00’ records) that IMS treats as checkpoint records. If you use the IMS HD Reorganization Reload utility to reload the database, IMS ignores these records. If you process the HD Unload data set as a sequential data set, ensure that the processing program recognizes the BMC X’00’ header records.

- IMS zeros the pad byte added to make the segment length even; the Unload function does not.

- The HD Unload data set that is produced by the Unload function is valid input to any IMS reload program. However, if you request the BMC format SHORT or XSHORT for the record prefixes, only the Reload function can use the resulting HD Unload data set to reload the database. Any user programs that do not process the prefix can read the data set regardless of prefix format. For more information, see "DFSURGU1" on page 425.
Performance considerations

This section discusses how you can tune the Unload function to perform most efficiently.

Buffers

The Unload function uses one set of buffer pools. These buffer pools are allocated and dynamically tuned within the Unload function. You cannot specify keywords to tune buffers; however, you can use the BUFSPACE keyword to specify the maximum amount of buffer space to be used for the unload step. Each data set group in the database has its own buffer pool. The unit of allocation is always tracks. Storage for these buffer pools is obtained above the 16 MB line. A track full of blocks is read into the pool at one time; if this method is not efficient, one block at a time is read into the pool.

VSAM KSDS buffers

If the database data set is a VSAM KSDS, you can specify the AMP operand in the data set’s DD statement. You can also specify the BUFNI/BUFND or BUFSP subparameters of AMP. The Unload function honors the user-specified values if they are adequate. The Unload function uses BUFNI=16 and calculates the minimum BUFND value to be equal to the number of CIs in a cylinder plus one. These values control the size of the VSAM buffer pool for the data set unless you specified larger values.

DFSURGU1 buffers

QSAM is used to write the HD Unload data set to the data set that is specified in the DFSURGU1 and DFSURGU2 DD statements when you are writing to disk, unless the SYNCTAPE keyword is specified. The minimum number of buffers that are allocated to each data set is five. You can increase the allocation by using the BUFNO parameter in the DCB operand for these DD statements. If the database contains large segments, you might want to specify more than five buffers.

The Unload function calculates the LRECL and BLKSIZE values. The LRECL value is equal to the larger of the following values:

- largest segment data length plus the prefix size
- IBM header record size
The BLKSIZE value is the maximum size for the device type. If you specify a BLKSIZE value with the DCB JCL operand, it is used. If the BLKSIZE you specify is too small, the Unload function calculates a larger value automatically.

The Unload function uses EXCP to write the HD Unload data sets when SYNCTAPE is specified. One set of buffers is used for the DFSURGU1 and DFSURGU2 data sets. You can control the number of buffers allocated with the BUFNO parameters of the DCB operand of the DFSURGU1 DD statement. The default is 10. The sum of the DCBNO values is used if the number of buffers is 10 or more.

Options

The Unload function options shown in Table 28 can affect performance. For more information, see “Commands and keywords overview” on page 388.

Table 28 Unload function options that affect performance

<table>
<thead>
<tr>
<th>Option</th>
<th>Effect on CPU usage</th>
<th>Reduces EXCPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFSURGU1(SHORT) or DFSURGU1(XSHORT)</td>
<td>reduces</td>
<td>yes</td>
</tr>
<tr>
<td>EXPAND(NO)</td>
<td>reduces</td>
<td>yes</td>
</tr>
<tr>
<td>LPCK(NO)</td>
<td>reduces</td>
<td>yes</td>
</tr>
<tr>
<td>REPORTS(IOSTATS(YES))*</td>
<td>increases</td>
<td>no</td>
</tr>
<tr>
<td>REPORTS(KEYS) or REPORTS(KEYS(nnn))</td>
<td>increases</td>
<td>no</td>
</tr>
<tr>
<td>REPORTS(RCDDIST(YES))*</td>
<td>increases</td>
<td>no</td>
</tr>
</tbody>
</table>

* indicates default value. REPORTS(KEYS(10)) is the default value.

The reduction in EXCPs for LPCK(NO) is significant.

DB Scan function

The DB Scan function is a functional replacement for the IMS Database Scan utility. When you run the DB Scan function, the OS/VS Initiator loads the batch region controller (DFSRRC00). The region controller loads and transfers control to the Unload function. You can use DB Scan any time that you run the Unload function as an independent utility.
When the Unload function receives control, it determines the values to use for the user-controlled options as explained in “How the utilities build options” on page 385. The Unload function then builds control blocks, opens data sets, and begins to scan the database.

The Unload function retrieves input segments from the database and builds the appropriate work file 1 records. It uses standard OS/VS access methods rather than DL/I to scan the database.

When the Unload function scans the database, it accumulates various statistics (for example, segment placement information and database record sizes). At the end of the scan, the Unload function closes the data sets and generates the statistical reports. If you provide a PDX data set, the information for the reports is also stored in the PDX. You can view or print the reports at any time using the ISPF interface.

**NOTE**

Reports pertaining to HALDBs and the IMS/ESA Partition Support Product with a partitioned primary index cannot be stored in the PDX. Report information for HALDBs is in the SYSOUT statement.

The Unload function works with DBRC in the same way as the IMS Database Scan utility does.

**NOTE**

You cannot scan HALDBs using the DB Scan function.

Figure 37 on page 189 shows the general system flow when you run the DB Scan function.
JCL requirements

Figure 38 shows that the JCL requirements for the DB Scan function are similar to those for the IMS Database Scan utility (DFSURGS0). Depending on how you installed the Unload function, you might need to modify the JCL by adding a STEPLIB DD statement.

Figure 38  Sample JCL to run the DB Scan function (part 1 of 2)

```plaintext
//UNLOAD EXEC PGM=DFSRRC00,PARM='ULU,DFSURGS0'
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//DFSURCDS DD DISP=SHR,DSN=your.DFSURCDS
//database DD DISP=OLD,DSN= dataSetName
```
An explanation of the JCL follows:

**EXEC**

You must specify PGM=DFSRRRC00,PARM='ULU,DFSURGS0'. You can add the IMSID parameter to override the IMSID specified in the IMS RESLIB.

**STEPLIB DD**

Optional. For more information, see “STEPLIB DD” on page 378.

**IMS DD**

Required. Describes the library, typically IMSVS.DBDLIB, containing the DBDs that describe the databases being scanned.

**DFSURCDS DD**

Required. For more information, see “DFSURCDS DD” on page 359.

**database DD**

Required. Defines the data sets that contain the database being scanned. You must supply one DD statement for each data set appearing in the DBD. The ddnames must match the ddnames specified in the DBD.

**BMCMSG DD**

Optional. For more information, see “BMCMSG DD” on page 353.
DFSURWF1 DD

Required. For more information, see “DFSURWF1 DD” on page 364.

**NOTE**

The Unload function uses the BMC DFSURWF1 work file generator to build the work file records. If you scan the database with the Unload function and then use the IBM DFSURWF1 work file generator when loading it, you must specify the DCB information on the SORTIN or DFSURWF1 DD statement in the JCL for PREFIX RESOLUTION PLUS or the IMS Prefix Resolution utility.

PRPURWF1 DD

Optional. For more information, see “PRPURWF1 DD” on page 375.

BMCPRINT DD

Optional. For more information, see “BMCPRINT DD” on page 354.

PLUSIN DD

Optional. The DBSCAN command is required. The following keywords are optional:

- DESCCODE
- DYNALLOC
- IMSID
- LPCK
- MONITOR
- MONUSERS
- PTRERROR
- ROUTCODE
- USEINDEX

For information about the tasks these keywords accomplish, see Chapter 14, “User-controlled options.” For information about how to specify the keywords, see “PLUSIN control statement syntax” on page 404.

PDX DD

Optional. For more information, see “PDX DD” on page 371.

SYSUDUMP DD

Recommended. For more information, see “SYSUDUMP DD” on page 380.
Database Selection facility

The Database Selection facility (DBSF), available in the Unload function, allows you to retrieve selected database records from a database and to print them or write them to an HD Unload data set. You can use the DBSF to create test databases or to print or unload specific database records.

To process a partial subset, use the DBSF. You must know the boundaries of the subsets. For example, if you want to unload the last part of subset 1 and the first part of subset 2, you can use the DBSF to extract that subset using the Unload function.

For HDAM and PHDAM databases, you must know the starting and ending relative block number for each subset. For HIDAM and PHIDAM databases, you must supply the key ranges for each subset.

NOTE
Partition selection routines are not automatically invoked. For HALDBs with partition selection routines, you must supply the appropriate ranges for each partition.

JCL requirements

The JCL requirements are the same as those for any DL/I batch program. In the Unload function, the DBSF does not use the IMS region controller services. Several additional DD statements are required to provide selection information and define the output data sets. Figure 39 shows sample JCL.

Figure 39  Sample JCL for Database Selection facility

```plaintext
//SELECT EXEC PGM=DFSRRC00,PARM='DLI,ULPDBSEL,psbname'
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.PSBLIB
// DD DISP=SHR,DSN=IMSVS.DBDLIB
//RECON1 DD DISP=SHR,DSN=your.recon1 <===for HALDB only; required
//RECON2 DD DISP=SHR,DSN=your.recon2 <===for HALDB only; required
//RECON3 DD DISP=SHR,DSN=your.recon3 <===for HALDB only; required
//DI21PART DD DISP=SHR,DSN=DI21PART
//DBSPRINT DD SYSOUT=*  
//BMCMSG DD SYSOUT=*  
//DFSURGU1 DD DISP=(,PASS),DSN=user.unload,UNIT=SYSDA,SPACE=(TRK,(2,1))
//PLUSIN DD *
DBSELECT BEGIN(100) END(199) SKIPFREQ(5)
```
The PSB specified in the PARM parameter of the EXEC statement must contain a PCB for the database being read. It must be the first DB PCB in the PSB, unless you specify a PCB with the PCB or PCBNAME keyword of the DBSELECT command. The PCB must reference a physical DBD; no logical DBDs are allowed. If the PCB does not define all segments in the database record, the DBSF unloads only the segments defined in the PCB. PROCOPT is not used.

Specify the database selection criteria through commands in the PLUSIN control statement data set. The DBSF does not access the PDX.

**DBSF keywords**

Each control statement consists of a DBSELECT command and its keywords. You can use more than one DBSELECT command to specify additional ranges of records to be unloaded from a DBD. Table 29 lists the valid keywords.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
</table>
| BEGIN   | For HDAM and PHDAM databases, BEGIN can represent one of the following values:  
  - a numeric value that specifies the RBN. The value must fall within the RAA. The first block, RBN 1, is the bitmap block. The unload begins at the first RAP in the specified RBN. If you do not specify a starting value, a value of 1 is used. If the PART keyword is not specified for PHDAM databases, the RBN is used for all partitions.  
  - the key of a single root. The exact, complete root key must be specified with the BEGIN keyword. If you specify BEGIN(rootKey), you cannot specify the COUNT, END, SKIP, or SKIPFREQ keyword.  
For HIDAM, PHIDAM and HISAM databases, BEGIN must be in character or hexadecimal format. A hexadecimal string must contain an even number of hexadecimal characters, enclosed in single quotes and preceded with the character X ('Xxxxx'). A character string is enclosed in single quotes and preceded by the character C ('Ccccc'). To specify a generic key, use a key value length less than the root’s key length. If a generic starting key is specified, it is padded with X’00’s. The unload begins at the first root with a key equal to or greater than the specified starting value. If you do not specify a starting value, the unload begins at the beginning of the database. |
| COUNT   | The COUNT keyword allows you to control the number of database records selected that lie within the BEGIN and END range. The value specified can be any number up to eight digits. If you specify BEGIN(rootKey), the COUNT keyword is not allowed. |
## JCL requirements

### Table 29  DBSELECT command keywords (part 2 of 2)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>END</td>
<td>For HDAM and PHDAM databases, END must be a numeric value that specifies an RBN within the RAA. The unload terminates after the last RAP in the specified block. If END is not specified, the last block in the RAA is used. If you specify BEGIN(rootKey), the END keyword is not allowed. If the PART keyword is not specified for PHDAM databases, the RBN is used for all partitions. For HIDAM, PHIDAM, and HISAM databases, END must be in character or hexadecimal format. A hexadecimal string must contain an even number of hexadecimal characters, enclosed in single quotes and preceded with the character X (X’xxxx’). A character string is enclosed in single quotes and is preceded by the character C (C’cccc’). A generic key is specified by using a key value length less than the root’s key length. If a generic ending key is specified, it is padded with XFF’s. The unload terminates before processing the first root with a key greater than the ending value. If no ending value is specified, the unload continues to the end of the database.</td>
</tr>
<tr>
<td>EXPAND</td>
<td>The EXPAND keyword allows you to control whether to expand the segment or leave it in compressed format. You can specify the EXPAND keyword only in the first DBSELECT command. The accepted values are YES and NO. The default is YES.</td>
</tr>
<tr>
<td>LPCK</td>
<td>The LPCK keyword allows you to not extract the LPCK if the database contains logical child segments whose LPCKs are not physically stored in the segment (they are virtual). You can specify the LPCK keyword only in the first DBSELECT command. The accepted values are YES and NO. The default is YES.</td>
</tr>
<tr>
<td>PART</td>
<td>The PART keyword identifies the PHDAM or PHIDAM database partition to be unloaded. Each partition requires a separate DBSELECT command. You can specify only one partition with the PART keyword. The value must be a valid partition name.</td>
</tr>
<tr>
<td>PCB</td>
<td>The PCB keyword allows you to specify which PCB within the PSB is to be used. This keyword can appear only in the first DBSELECT command. The PCB and PCBNAME keywords are mutually exclusive. If you do not specify PCB or PCBNAME, the first DB PCB is used. You can use the PCB keyword to specify a number or a DBD name. If you specify a PCB number, it must identify the DB PCB. For example, if the first DB PCB is specified PCB(1) and a DBD name is specified, the first PCB containing the specified DBD name is used.</td>
</tr>
<tr>
<td>PCBNAME</td>
<td>The PCBNAME keyword provides an alternate method of specifying which PCB within the PSB is used. You can specify the PCB by the PCB name assigned to it in the label field or the PCBNAME keyword of the PCB statement. The PCBNAME keyword can appear only in the first DBSELECT command. The PCBNAME and PCB keywords are mutually exclusive. If you do not specify the PCB or PCBNAME keyword, the first DB PCB is used.</td>
</tr>
<tr>
<td>PRINT</td>
<td>The PRINT keyword allows you to request that the output be in print format. The accepted values are NO, YES, and HEX. The YES and HEX values print the data and write it to DFSURGU1. If you want a print data set only, specify DFSURGU1 DD DUMMY and PRINT(YES) or PRINT(HEX). The default is NO. Print is written to a DBSPRINT DD.</td>
</tr>
<tr>
<td>SKIP</td>
<td>The SKIP keyword allows you to skip over a specified number of database records before the retrieval process. This initial skipping occurs after position in the database is established with the BEGIN keyword. If you specify BEGIN(rootKey), the SKIP keyword is not allowed.</td>
</tr>
<tr>
<td>SKIPFREQ</td>
<td>The SKIPFREQ keyword allows you to skip over a specified number of database records after retrieval of a database record. For example, SKIPFREQ(1) retrieves every other database record. SKIPFREQ(2) retrieves every third database record. The value can be any number up to five digits. If you specify BEGIN(rootKey), the SKIPFREQ keyword is not allowed.</td>
</tr>
</tbody>
</table>
Examples

The following examples demonstrate how to use the DBSELECT command:

**Example 1:**
The following example shows the selection of records through a PCB (DI21PART). The DBSF selects all database records accessible through the DI21PART PCB.

```
DBSELECT PCB(DI21PART)
```

**Example 2:**
The following example shows the selection of 100 records in a HIDAM or HISAM database. The DBSF selects and prints the first 100 records whose keys are in the range that begins with the characters ADAMS and ends with the characters SMITH.

```
DBSELECT BEGIN(C'ADAMS') END(C'SMITH') -
    COUNT(100) PRINT(YES)
```

**Example 3:**
The following example shows the selection of records in a HIDAM, PHIDAM, or HISAM database. The DBSF selects all records with keys in the range of X'00000000' through X'0000FFFF'.

```
DBSELECT BEGIN(X'00000000') END(X'0000FFFF')
```

**Example 4:**
The following examples show the selection of records in a HIDAM, PHIDAM, or HISAM database. The DBSF selects all records with keys in the range beginning with 4444 and ending with 7777. The two examples illustrate the use of a character string and a hexadecimal string.

```
DBSELECT BEGIN(C'4444') END(C'7777')
```

```
or
```
```
DBSELECT BEGIN(X'F4F4F4F4') END(X'0000FFFF')
```

**Example 5:**
The following example shows the selection of records in an HDAM database. The DBSF selects every sixth record in the blocks, beginning with RBN 100 and ending with RBN 199.

```
DBSELECT BEGIN(100) END(199) SKIPFREQ(5)
```
Example 6:
The following example shows the selection of one record in an HDAM database. The DBSF selects the record with the root key of ADAMS. The record is printed.

```
DBSELECT BEGIN(C'ADAMS') PRINT(YES)
```

Example 7:
The following example shows the selection of all records in a partitioned HDAM database with 1000 blocks in each partition beginning at RBN 950 in the first partition and ending at RBN 50 in the second partition.

```
DBSELECT BEGIN(950) END(1050)
```

Example 8:
The following example shows the selection of all records in a PHDAM database. The DBSF selects all records in partitions PAIRA and PAIRB beginning with RBN 2 and ending with RBN10 in both partitions.

```
DBSELECT BEGIN(2) END(10) PART(PAIRA)
DBSELECT BEGIN(2) END(10) PART(PAIRB)
```

Output

The output of the DBSF can be a print data set, an HD Unload data set, or both. The print output contains the following information for each segment that is retrieved:

- name
- RBA
- concatenated key
- prefix
- data

The output is directed to the BMCMSG (messages) and BMCPRINT (reports) data sets. The HD Unload output is an IMS-compatible HD Unload data set that can be used as input to the Reload function or the IMS HD Reorganization Reload utility. You must specify a DFSURGU1 DD statement for the HD Unload output.

Figure 40 on page 197 shows a sample of the printed output from the DBSF when you specify PRINT(YES).
Figure 40  Database Selection facility sample output (printed format)

1THU  5/23/96 144  BMC SOFTWARE, INC.  10:45

UNLOADPLUS/EP

SELECTION CRITERION:

START KEY=C'..............'
X'0000000000000000'  
END KEY=C'..............'
X'FFFFFFFFFFFFFFF'

>SEG=CFROOT  RBA=0000F004  CKEY=00...CONTROL01
PREFIX:0100 0000F14C 00000000 00000000 00000000 00000000 00000000 00000000 
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 
DATA:(00...CONTROL01X 990639416 999130204 )
  ( 050184 001310010378 000000000000000000
  )

>SEG=CFROOT  RBA=0000F14C  CKEY=00ABBREY084.000
PREFIX:0100 0000F2F8 0000F004 00000000 0000F294 00000000 00000000 00000000 00000000 
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 
00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 
DATA:(00ABBREY084.000 BOBBY ABBREY 554378765 003 1285 128)
  (5122385 X 007062 554378765
  )

SEG=CFADDR  RBA=0000F294  CKEY=00ABBREY084.0001
PREFIX:0300 0000F2C6
DATA:(AD11234 S MAIN ST )

SEG=CFADDR  RBA=0000F2C6  CKEY=00ABBREY084.0002
PREFIX:0300 00000000
DATA:(AD2PHOEINX ..AZ..85202)

SELECTED DB RECORDS=2

DB SELECTION COMPLETE
Figure 41 shows a sample of the printed output from the DBSF when you specify PRINT(HEX).

Figure 41  Database Selection facility sample output (hexadecimal format) (part 1 of 2)
The Unload function API lets a DL/I application program use the product’s fast database retrieval facilities. For batch DL/I programs that primarily read large portions of a database sequentially, using the API can significantly reduce elapsed time and CPU usage.

The API is not a replacement for full-function DL/I; it is an extension of the unload process. The API rapidly retrieves segments that are in relatively sequential order.

Using the API speeds the sequential retrieval of large numbers of segments, but you should not use it to retrieve only a few thousand segments. It optimizes processing for GN calls with unqualified segment search arguments (SSAs) and for GNP calls.
Unless you have several thousand GN calls for every GU call, do not invoke the API. If your application program requires full-function DL/I-type processing, use DL/I.

The API assumes that it has *read exclusive* control of the database (as any unload utility does during reorganization). You can run an API job while other jobs are updating the database; however, a BMC90212 message or U4094 abend might be issued, indicating that the database contains pointer errors when none actually exist. (In a similar situation, IMS issues an 85x abend or a GG status code.) The abend occurs when IMS writes an updated block to DASD while a related updated block is still in the IMS buffer pool and the API reads the updated block from DASD. This can be called a *temporary* pointer error.

The API supports GOx PROCOPTs and might return a GG status code. PROCOPT=GOx allows you to access segments even if they are locked. This access to a locked segment might result in a temporary pointer error. If you specify PROCOPT=GOx, the API returns a GG status code rather than causing the job to abend. Any application program that is using PROCOPT=GOx should contain logic to deal with this situation.

Be aware of potential data integrity exposures that can occur when you use PROCOPT=GOx. If data integrity is required, do not use PROCOPT=GOx.

The API is transparent to the application program. Standard DL/I call facilities are used. You do not need to recompile or relink-edit the application program. Implement the API with execution-time JCL. DL/I calls using the Application Interface Block (AIB) interface are supported.

**System flow**

Figure 42 on page 201 shows the system flow of the API.

---

**NOTE**

The API does not access the PDX.
The API uses the following process:

1. Because the API uses standard DL/I batch JCL, the IMS region controller (DFSRRC00) is invoked by the MVS Initiator, which initializes the region for batch DL/I. BMC provides a module with an alias of DFSPR000. When the BMC-provided DFSPR000 module receives control for each DL/I call, it analyzes and intercepts DL/I calls to the PCBs you designated it to handle (through the PCB and/or PCBNAME keyword). All other DL/I calls to other PCBs are passed to the IBM-provided DFSPR000 module.

2. When the application program makes a DL/I call, the BMC-provided DFSPR000 module receives control.
3. The BMC-provided DFSPR000 module checks the PCB:

- If the PCB is to be processed by the Unload function, control passes to the BMC product.
- If the PCB is not to be processed by the Unload function, control passes to the IBM-provided DFSPR000 module and the call is serviced by DL/I.

**JCL requirements**

To allow a DL/I batch program to use the API, add several DD statements to the batch program’s run-time JCL. No special PCB or PSB is required. The application program uses the standard DL/I call interface. You do not need to recompile or relink the application program.

Figure 43 shows sample JCL to read a database by using a DL/I retrieval program with the API.

**Figure 43  Sample JCL to run a DL/I retrieval program using the API**

```plaintext
//USERREAD  EXEC  PGM=DFSRRC00,PARM=’DLI,readpgm,getpsb’
//STEPLIB  DD  DISP=SHR,DSN=bmc.xxx.load  <===  required
  //  DD  DISP=SHR,DSN=your.program.load
  //  DD  DISP=SHR,DSN=IMSVS.RESLIB
  //DFSRESLIB  DD  DISP=SHR,DSN=IMSVS.RESLIB
  //  DD  DISP=SHR,DSN=IMSVS.PSBLIB
  //  DD  DISP=SHR,DSN=IMSVS.DBDLIB
  //RECON1  DD  DISP=SHR,DSN=your.recon1  <===  for  HALDB  only;  required
  //RECON2  DD  DISP=SHR,DSN=your.recon2  <===  for  HALDB  only;  required
  //RECON3  DD  DISP=SHR,DSN=your.recon3  <===  for  HALDB  only;  required
  //DFSVSAMP  DD  DISP=SHR,DSN=IMSVS.PROCLIB(VSAM01)
  //IEFRDER  DD  DUMMY
  //BMCPRINT  DD  SYSOUT=*  
  //BMCMSG  DD  SYSOUT=*  
  //PLUSIN  DD  *  <===  required
USERREAD  PCB(dbdbName  |  pcbNumber)
//
//database  DD  DISP=SHR,DSN=your.database.dataSet
```

An explanation of the JCL follows:

**EXEC**

Required. Specify PGM=DFSRRC00,PARM=’DLI,readpgm,getpsb’, where readpgm is the name of the application program and getpsb is the PSB name. You can use DBB rather than DLI in the PARM operand.
STEPLIB DD

Required. Must point to BMC load library. If several DD statements appear in the STEPLIB statement, BMC load library must be ahead of the IMS RESLIB.

**NOTE**
All libraries in the STEPLIB concatenation must be APF authorized, or you must have started the DBUSS.

DFSRESLB DD

Required. Describes the library that contains the IMS RESLIB.

IMS DD

Required. Must point to the DBD library and the PSB library that contain the DBDs and PSBs for the PCBs that are being serviced by the Unload function.

RECONx DD

Required for HALDBs. Defines the DBRC RECON data sets.

DFSVSAMP DD

Required. Supplies the buffer parameters that are not handled by the API.

IEFRDER DD

Follows the standard requirements for IEFRDER DD statements in any IMS batch job.

APITRACE DD

Optional. Contains output from the TRACE command. The APITRACE data set is dynamically allocated if it is needed.

PLUSIN DD

Required. Contains the commands and keywords for the API to use. Valid keywords are described with each command.

The USERREAD command is required for the API to service DL/I calls. The following commands are optional:

- UNLOAD
- TRACE
USERREAD command

The USERREAD command specifies the PCBs that the API will service. The USERREAD command can also provide some debugging information. The following keywords are valid with USERREAD:

Table 30 USERREAD command keywords (part 1 of 2)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCB</td>
<td>The PCB keyword allows you to specify one or more PCBs within the PSB that the API processes. You can specify the PCB by number or DBD name. When you specify a HALDB DBD name, all PCBs referencing that HALDB database are under API control. When you specify a non-HALDB DBD name, the first PCB that refers to that DBD is under API control. When you specify a PCB number for a non-HALDB database that is in the PCB, only that PCB is under API control. A maximum of 40 values are allowed for this keyword.</td>
</tr>
<tr>
<td>PCBNAME</td>
<td>The PCBNAME keyword provides an alternate method of specifying one or more PCBs within the PSB that the API processes. You can specify the PCB by the PCB name assigned to it in the label field or the PCBNAME keyword of the PCB statement. If the referenced database is a non-HALDB database, only the selected PCB is under API control. A maximum of 40 values are allowed for this keyword.</td>
</tr>
</tbody>
</table>
| IMSFIRST | The IMSFIRST keyword is a debugging tool that allows you to see the results of the call as made by DL/I first and the same call as made by the Unload function. Specify Y or N. The default value is N. If you use this keyword, the PCBs must specify a G-type PROCOPT. 

Note: The USERREAD command with the IMSFIRST keyword is not allowed with the Snapshot Copy function. |
| PRINT    | The PRINT keyword prints the DL/I call, the PCB information, and the contents of the user I/O area. If you want all information (CALL, PCB, and IOAREA), specify PRINT(ALL); the information prints to the PLUSLIST DD statement. If you omit the PRINT keyword, the default value is NO for the USERREAD command. |
| SNAPDBD  | The SNAPDBD keyword is valid only if you have EXTENDED BUFFER MANAGER for IMS (XBM for IMS) or the SNAPSHOT UPGRADE FEATURE (SUF for IMS) installed. XBM Snapshot allows you to take a Snapshot Copy of a database. Although the Snapshot process copies all related databases, the API is an extension of the Unload process (DFSURGU0) and can only process a physical DBDs. For more information, see “PSB conditions” on page 208. |
| XBMSSID  | When you specify the SNAPDBD keyword, you must specify the name of the XBM subsystem with the XBMSSID keyword. The XBM subsystem intercepts any updates made to an XBM-registered database, and it preserves the pre-image of the database. The API includes the pre-image in the Snapshot Copy instead of the corresponding block read from the database. |
Chapter 6  Unload function

Example 1:
The following example shows how to specify keywords for the USERREAD command when you use the PCB keyword:

```
USERREAD PCB(dbdName,...|pcbNumber,...) +
IMSFIRST(Y|N) +
PRINT(ALL|CALL,PCB,IOAREA)
```

Example 2:
The following example shows how to specify keywords for the USERREAD command when you use the PCBNAME keyword:

```
USERREAD PCBNAME(pcbName,...,pcbName) +
IMSFIRST(Y|N) +
PRINT(ALL|CALL,PCB,IOAREA)
```

### TRACE command

The Trace/Debug facility provides a debugging tool for DL/I application programmers. When you specify the PRINT keyword, the Trace/Debug facility lists an application program’s DL/I call and the resulting PCB information. (The USERREAD PRINT option displays the results of the Unload function call.) The TRACE and USERREAD commands cannot process the same PCBs in the same job step.

Implement the Trace/Debug facility by adding the following DD statements to your run-time JCL:

- STEPLIB DD statement (must point to BMC load library)
- PLUSIN DD statement
- APITRACE DD statement (optional; will be dynamically allocated if needed)
The UNLOAD function API load library must appear before the IMS RESLIB in the STEPLIB concatenation.

Specify the TRACE command and its keywords by using the following format:

```
TRACE PCB(dbdName|PCBNumber) PRINT(ALL|CALL,PCB,IOAREA)
```

The following keywords are valid with TRACE:

**Table 31  TRACE command keywords**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCB</td>
<td>The PCB keyword allows you to specify one or more PCBs within the PSB that the API processes. You can specify the PCB by number or DBD name. When you specify by name, the API processes the first matching occurrence of the DBD. A maximum of 40 values are allowed for this keyword.</td>
</tr>
<tr>
<td>PCBNAME</td>
<td>The PCBNAME keyword provides an alternate method of specifying one or more PCBs within the PSB that the API processes. You can specify the PCB by the PCB name assigned to it in the label field or the PCBNAME keyword of the PCB statement. A maximum of 40 values are allowed for this keyword.</td>
</tr>
<tr>
<td>PRINT</td>
<td>The PRINT keyword prints the DL/I call, the PCB information, and the contents of the user I/O area. If you want all information (CALL, PCB, and IOAREA), specify PRINT(ALL). The information prints to the APITRACE DD statement. If you omit the PRINT keyword, the default value is ALL.</td>
</tr>
</tbody>
</table>

**Example:**
The following are examples of the TRACE command and keywords:

```
TRACE PCB(dbdName) PRINT(CALL,IOAREA)
TRACE PCB(1,3) PRINT(ALL)
```

**UNLOAD command**

The UNLOAD command is valid with the USERREAD command for compatibility with UNLOAD PLUS. The UNLOAD command allows you to use the USEINDEX keyword with the API. If you do not specify the USEINDEX keyword, the default is NO.

Because the Unload function API supports the PROCSEQ that is specified in the PCB, the USEINDEX keyword is not required to process a database using the secondary index.
If you specify USEINDEX(indexName), the secondary index that you specify must be associated with the database that was selected with the PCB or PCBNAME keyword on the USERREAD command. If multiple PCBs were selected, all those PCBs must be associated with the database that has the secondary index.

**Restrictions**

Because the Unload function is a high-speed sequential database read program, it can service DL/I get-type calls. For best performance, limit the application program calls to GN and GNP calls. GU calls are supported; however, IMS typically would service those calls as efficiently as the Unload function. The API does not request an implicit verify for VSAM databases.

The API is not a complete replacement for full-function DL/I.

**DL/I conditions**

The API works under the following DL/I conditions:

- The API supports GU, GN, GNP, OPEN, and CLSE DL/I calls.
- The API supports GU calls, but it does not optimize them.

Unless you have several thousand GN calls for every GU call, do not invoke the API. If your application program requires full-function DL/I processing, use DL/I.

- GN processing does not support fully qualified SSAs:
  
  — HDAM root segments should be retrieved with an unqualified SSA or with a GU call.

  — Lower-level processing for HDAM and HIDAM should use GNP calls or an unqualified SSA.

  The calls will be processed, but positioning is not fully compatible with IMS when key names and values are included. Results are unpredictable. Do not invoke the API.

- If the PCB contains an unsupported call, an AD status code is returned to the application program.

- GHxx calls are supported. Only the GET (data retrieval) portion of the call is performed. No enqueue is placed on the data for the delete, insert, or replace.

- GA and GK status codes are not generated; blanks are issued instead.
The acceptable command codes are D (path calls) and P (parentage). If the PCB contains an unsupported command code, an AJ status code is returned to the application program.

No Boolean operators are allowed in SSAs. An AJ status code is returned.

The API cannot run as a BMP.

HSAM databases cannot be read.

Index databases cannot be processed as stand-alone databases. Control is returned to DL/I for processing.

PL/1 programs must adhere to IMS considerations for PL/1 (described in the PL/1 Programmer’s Guide):

— Compile with NOSPIE and NOSTAE options.
— Force PL/1 program termination through an operating system abend request (modified IBMBEERA module for PL/1 release 5).

If the API is controlling a PSB that references a HALDB database, then all PCBs for the database must be controlled by the API.

**PSB conditions**

The following PSB restrictions apply:

The DBD that is referenced by the PCB must be a physical DBD; it cannot be a logical DBD. The API does not retrieve virtual logical child segments.

POS=M is not supported and is ignored.

Segments in the PCB must be in hierarchical order.

If an API job is running while the database is being updated, the PSB must specify the appropriate GOx PROCOPT and the application program must handle the potential GG status code. See the description of this scenario in “API” on page 199.

PROCSEC is not valid with HALDBs.
HSSR conversion utility

You can use the API to trap application program HSSR calls and pass them to IMS.

The application program must be link-edited to replace the HSSR language interface modules (FABHHSSR and FABPLI) with ULPHSRL1.

ULPHSRL1 link-edits with unresolved addresses for ASMTDLI, CBLTDLI, and PLITDLI. This is necessary to avoid shipping IMS code with this module. However, those addresses will be resolved when you relink the application program.

Sample JCL is in the sample library member ULPHSRL1.

The JCL shown in Figure 44 illustrates the procedure for relinking the application program.

**Figure 44  JCL to relink HSSR application programs**

```plaintext
//LKED EXEC PGM=IEWL,REGION=0M,
//       PARM=(XREF,LET,LIST,REUS)
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=&&SYSUT1,UNIT=VIO,SPACE=(1024,(50,20))
//USERLIB DD DISP=SHR,DSN=customer.pgmlib
//BMCLIB DD DISP=SHR,DSN=bmc.xxx.load
//SYSLIB DD DISP=SHR,DSN=customer.pgmlib
//SYSLIB DD DISP=SHR,DSN=IMSVS.RESLIB
//SYSLMOD DD DISP=SHR,DSN=customer.pgmlib
//SYSLIN DD *,
REPLACE FABHHSSR(ULPHSRLI)
REPLACE HSSR(ULPHSRLI)
REPLACE FABHPLI(ULPHSRLI)
INCLUDE USERLIB(APITST10)
INCLUDE BMCLIB(ULPHSRLI)
INCLUDE SYSLIB(DFSLLIB)
ENTRY DLITCBL <= depends on program language
NAME APITST10(R)
```

The JCL shown in Figure 45 illustrates the change required to the application’s run JCL.

**Figure 45  JCL changes required for HSSR application programs (part 1 of 2)**

**BEFORE:**

```plaintext
//HSSRSTEP EXEC PGM=FABHX034,
//      PARM=('DFSRRC00/DLI',APITST10,SOMEPSB, ... etc ...)
```

**AFTER:**

```plaintext
//ULPSTEP EXEC PGM=DFSRRC00,
//      PARM=('DLI',APITST10,SOMEPSB, ... etc ...)
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
//      DD DISP=SHR,DSN=IMSVS.PGMLIB
//      DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
```
The Unload function API produces the following reports:

- Options in Effect (described on page 604)
- DBD Summary (described on page 592)
- Disassembled DBD (described on page 594)
- Disassembled PSB (shown in Figure 46)

**Figure 46  Disassembled PSB report**

```
PCB   TYPE=DB,DBDNAME=BA$HIDO0,KEYLEN=16,PROCOPT=A
SENSEG NAME=BA#AASEG,PARENT=0
SENFLD NAME=BA@AAKEY,START=5,REPL=N
SENSEG NAME=BA#BASEG,PARENT=BA#AASEG,PROCOPT=I
SENFLD NAME=BA@BAKEY,START=5,REPL=N
SENSEG NAME=BA#BDSEG,PARENT=BA#AASEG,PROCOPT=G
SENFLD NAME=BA@BDKEY,START=5,REPL=N
PSBGEN PSBNAME=APIBA$HP,LANG=COBOL,CMPAT=YES
END
```
Reload function

This chapter describes how to use the Reload function as a stand-alone utility. This chapter includes the following topics:

Overview .................................................. 212
JCL requirements ........................................ 213
   Non-parallel reload .................................. 214
   Parallel reload ...................................... 214
 DD statements ........................................... 215
 PLUSIN examples ...................................... 215
Keywords ................................................... 216
Global options modules ............................... 219
   Create using ISPF interface ....................... 220
   Create using batch job ............................. 222
Reports .................................................... 222
HD Reorganization Reload differences ............ 223
Performance considerations ......................... 224
   Changing the HDAM or PHDAM randomizer .......... 224
   Options ............................................. 225
Multivolume data sets ............................... 225
   VSAM data sets .................................... 225
   OSAM data sets .................................... 225
Empty database initialization ....................... 226
HD Sort utility .......................................... 227
   System flow ....................................... 229
   JCL requirements ................................ 229
   Example .......................................... 233
API ......................................................... 234
   System flow ....................................... 234
   JCL requirements ................................ 235
 USERLOAD command ............................. 237
 DL/I status codes ................................... 239
 Restrictions ........................................... 239
WF1 Split function ................................. 241
The Reload function (invoked with the RELOAD command) replaces the IMS HD Reorganization Reload utility (DFSURGL0). Figure 47 shows the system flow for the Reload function.

When you execute the RELOAD command, the first program to be executed is the BMC version of the batch region controller (DFSRRC00), which loads and transfers control to the Reload function.

When the Reload function receives control, it determines the values to use for the user-controlled options as explained in “How the utilities build options” on page 385.

The Reload function builds miscellaneous control blocks, opens data sets, and begins reloading the database. It retrieves input segments from the HD Unload data set in hierarchical order. If the key fields are not compressed, the Reload function checks the sequence of these segments. It uses standard OS access methods to build the database data sets and accumulates various statistics as it builds the database.
When the reload process completes, the Reload function closes the database data sets and generates various reports, which can be stored in the PDX. You can view these reports, and previous reports for the same database, any time through the ISPF interface.

**NOTE**
Reports pertaining to HALDBs or the IMS/ESA Partition Support Product with a partitioned primary index cannot be stored in the PDX. Report information for HALDBs is located in the SYSOUT statement.

Secondary index records are written to the DFSURWF1 data set.

If the database contains logical relationships, the Reload function builds the work file 1 records needed by the IMS Prefix Resolution utility or the BMC Prefix Resolution/Update function.

If you specify the PRPURWF1 DD statement, the Reload function separates the logical relationship records from secondary index records and writes the logical relationship work file records to the PRPURWF1 data set. The PRPURWF1 data set can be processed only by the Prefix Resolution/Update function. Subsequent processing of logical relationship records and secondary index records can be performed simultaneously, reducing elapsed time.

The Reload function works with DBRC in a similar manner to the IMS HD Reorganization Reload utility. The same exclusive authorization is done, and a NOTIFY.REORG is issued. Additionally, the utility turns on the prohibit authorization action at the beginning of the Reload function and turns it off at the termination of the Reload. The prohibit authorization action prevents an incompletely loaded database from being placed online.

**JCL requirements**

As described in the following sections, the JCL requirements for the Reload function are similar to the IMS HD Reorganization Reload utility JCL requirements.
Non-parallel reload

Figure 48 shows the JCL requirements for a non-parallel reload.

**Figure 48  Reload function JCL (non-parallel reload)**

```plaintext
//RELOAD EXEC PGM=DFSRRC00,PARM='ULU,DFSURGLO,dbname'
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//RECON1 DD DISP=SHR,DSN=your.recon1  ===> for HALDB only; required
//RECON2 DD DISP=SHR,DSN=your.recon2  ===> for HALDB only; required
//RECON3 DD DISP=SHR,DSN=your.recon3  ===> for HALDB only; required
//DFSUINPT DD DISP=OLD,DSN=test.hdunld
//dbdds1 DD DISP=OLD,DSN=database.dataSetName.dsg1
//dbdds2 DD DISP=OLD,DSN=database.dataSetName.dsg2
//DFSURCDS DD DISP=OLD,DSN=test.cds
//DFSURWF1 DD DISP=(NEW,CATLG,DELETE),SPACE=(CYL,(10,10)),
//           DSN=test.dfswrf1
//PRPURWF1 DD DISP=(NEW,CATLG,DELETE),SPACE=(CYL,(10,10)),
//           DSN=test.prpurwf1
//BMCMSG DD SYSOUT=*  
//BMCPRINT DD SYSOUT=*  
//PLUSIN DD *  
//RELOAD
```

Parallel reload

Figure 49 shows the JCL requirements for a parallel reload.

**NOTE**
To reload a database in parallel mode, it must have been unloaded with the Unload function in parallel mode.

**Figure 49  Reload function JCL (parallel reload) (part 1 of 2)**

```plaintext
//RELOAD EXEC PGM=DFSRRC00,PARM='ULU,DFSURGLO,dbname'
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//RECON1 DD DISP=SHR,DSN=your.recon1  ===> for HALDB only; required
//RECON2 DD DISP=SHR,DSN=your.recon2  ===> for HALDB only; required
//RECON3 DD DISP=SHR,DSN=your.recon3  ===> for HALDB only; required
//DFSUIN01 DD DISP=OLD,DSN=test.hdunlid1
//DFSUIN02 DD DISP=OLD,DSN=test.hdunlid2
//DFSUIN03 DD DISP=OLD,DSN=test.hdunlid3
//dbds DD DISP=OLD,DSN=database.dataSetName
```
For more information about the DD statements, see Chapter 13, “JCL requirements.”

The following DD statement is specific to the Reload function RELOAD command:

**dbddsgn/dbds DD**

Optional; however, if you specify DYNALLOC(YES) and the data sets are registered to DBRC, the Reload function dynamically allocates these data sets. Do not include these DD statements if you specify DYNALLOC(YES). This DD statement defines the data sets containing the reloaded database. One statement of this type must be present for each data set appearing in the DBD that describes the database. The ddnames must match the ddnames in the DBD or be specified with the HALDB Partition Definition Utility (PDU).

For more information, see “Multivolume data sets” on page 225.

If the database data set is a VSAM data set, you can use the AMP parameter. For more information, see the discussion of VSAM KSDS buffers on page 186.

**PLUSIN examples**

The following examples show some possible keyword combinations:

**Example 1:**
The first occurrence of segments named SEG1 and SEG2 are placed near the root:

```
RELOAD SEGPOS((SEG1,1)(SEG2,1))
```
Example 2:  
The reports are sent to the LDPPRINT data set. A corresponding LDPPRINT DD statement can be included in the JCL. The root keys of the 25 largest records are listed on the Database Record Statistics report:

RELOAD LDPPRINT(LDPPRINT) REPORTS(KEYS(25))

Example 3:  
The flip-flop space search method (searching forward first) is used for an HDAM database:

RELOAD SCAN(+,-10)

Example 4:  
The reload database data sets are dynamically allocated:

RELOAD DYNALLOC(YES)

Keywords

The Reload function uses the RELOAD command. Table 32 lists valid keywords.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKUPDATE</td>
<td>(YES,STOPn)</td>
<td>(YES,STOP4094)</td>
<td>414</td>
<td>Specify whether to check the RECON data sets for update allocation records and what action the Reload function takes if any records exist.</td>
</tr>
<tr>
<td>COMPRESS</td>
<td>(YES), (NO)</td>
<td>(YES)</td>
<td>415</td>
<td>Compress segments during reload.</td>
</tr>
<tr>
<td>DESCNODE</td>
<td>(nn,...) max 10 entries</td>
<td>(7)</td>
<td>490</td>
<td>Specify descriptor codes used to route messages to the console.</td>
</tr>
<tr>
<td>DFSDBUX1</td>
<td>(YES), (NO)</td>
<td>(YES)</td>
<td>424</td>
<td>Specify whether to invoke the DFSDBUX1 user exit.</td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>(YES), (NO)</td>
<td>(NO)</td>
<td>430</td>
<td>Dynamically allocate database data sets using the data set names specified in the RECON data sets. For HALDBs, DYNALLOC(YES) is required; it is also the default.</td>
</tr>
<tr>
<td>FBFF</td>
<td>(n1,n2...n10)</td>
<td></td>
<td>434</td>
<td>Specify an override value for the free block frequency factor specified in the DBD.</td>
</tr>
<tr>
<td>Keyword</td>
<td>Accepted values</td>
<td>Default values</td>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------</td>
<td>----------------</td>
<td>------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FMTALL</td>
<td>(YES)</td>
<td>(NO)</td>
<td>438</td>
<td>Format all data sets.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FMTRAA</td>
<td>(YES)</td>
<td>(NO)</td>
<td>439</td>
<td>Format the HDAM or PHDAM root addressable area.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(YES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSPF</td>
<td>(n1,n2...n10)</td>
<td>95</td>
<td>441</td>
<td>Specify a percentage of free space to qualify a block as free.</td>
</tr>
<tr>
<td>HDAMFSPF</td>
<td>(IGNORE)</td>
<td>(NO)</td>
<td>443</td>
<td>Ignore the free space percentage factor.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDSORT</td>
<td>(YES)</td>
<td>(NO)</td>
<td>443</td>
<td>Invoke the HD Sort utility or the BMC RAP Sequencer.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(AUTO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDCAMS</td>
<td>(input,output)</td>
<td>none</td>
<td>464</td>
<td>Specify SYcin and SYSPRINT files for IDCAMS processing.</td>
</tr>
<tr>
<td></td>
<td>(input,output,Y)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(input,output,N)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(model)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ILDS</td>
<td>(AUTO)</td>
<td>(AUTO)</td>
<td>468</td>
<td>Specify to build or initialize Indirect List Data Sets.</td>
</tr>
<tr>
<td></td>
<td>(BUILD)</td>
<td>(INIT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMSID</td>
<td>(imsID)</td>
<td></td>
<td>470</td>
<td>Specify IMSID under which reports are stored in the PDX.</td>
</tr>
<tr>
<td>LDPPRINT</td>
<td>(ddname)</td>
<td>BMCPRINT</td>
<td>522</td>
<td>Specify the data set that will contain the Reload function reports and processing output.</td>
</tr>
<tr>
<td>MONITOR</td>
<td>(nnnnnnnn)</td>
<td>no automatic monitoring</td>
<td>490</td>
<td>Monitor job progress.</td>
</tr>
<tr>
<td>MONUSERS</td>
<td>(user01,...)</td>
<td></td>
<td>490</td>
<td>Send monitor messages to TSO users and BMCMSG.</td>
</tr>
<tr>
<td>OSAMINIT</td>
<td>(YES)</td>
<td>(NO)</td>
<td>499</td>
<td>Specify the reset of the high-used RBA for a database that has multi-volume OSAM data sets.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(YES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OSAMMAX</td>
<td>(4)</td>
<td>(8)</td>
<td>500</td>
<td>Specify the size limit of an OSAM database.</td>
</tr>
<tr>
<td>OUTDDX</td>
<td>(ddname(indexDB))</td>
<td>none</td>
<td>501</td>
<td>Specify the data set that will contain secondary index records.</td>
</tr>
<tr>
<td></td>
<td>(AUTO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OVFLONLY</td>
<td>(segname1,segname2,...)</td>
<td>none</td>
<td>503</td>
<td>Place specified segments in HDAM overflow area.</td>
</tr>
<tr>
<td></td>
<td>max 25 names</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 32  Reload function keywords (part 3 of 4)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PADCHAR</td>
<td>PADCHAR(x)</td>
<td>PADCHAR('x')</td>
<td>PADCHAR(&quot;x&quot;)</td>
<td>PADCHAR(C'x')</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>(YES)</td>
<td>(NO)</td>
<td>(YES)</td>
<td>504</td>
</tr>
<tr>
<td>PART</td>
<td>(ALL)</td>
<td>(partitionName, ...)</td>
<td>(nn,nn,...)</td>
<td>508</td>
</tr>
<tr>
<td>PDX</td>
<td>(OKAYnnn)</td>
<td>(STOPnnn)</td>
<td>(OKAY0004)</td>
<td>518</td>
</tr>
<tr>
<td>PDXPARMS</td>
<td>(userName)</td>
<td>Use the DBD name specified in the EXEC statement's PARM operand.</td>
<td>520</td>
<td>Override the DBD name member in the PDX.</td>
</tr>
<tr>
<td>REPORTS</td>
<td>See page 526</td>
<td>See page 526</td>
<td>526</td>
<td>Generate reports.</td>
</tr>
<tr>
<td>RMBYTES</td>
<td>(SEGSIZE)</td>
<td>(MINBYTES)</td>
<td>(MINBYTES)</td>
<td>529</td>
</tr>
<tr>
<td>ROUTCODE</td>
<td>(nn,...)</td>
<td>(2,7)</td>
<td>490</td>
<td>Specify routing codes to control destination of messages sent to the console.</td>
</tr>
<tr>
<td>SCAN (space search)</td>
<td>-nnn+mmm</td>
<td>(+nnn-nmm)</td>
<td>(+nmm),(+nmm)</td>
<td>531</td>
</tr>
<tr>
<td>SEGPOS</td>
<td>(segname1,n1, segname2,n2,...)</td>
<td>none</td>
<td>535</td>
<td>Specify which segment type to place near the root.</td>
</tr>
<tr>
<td>SEGPOSO</td>
<td>(ALL)</td>
<td>(segname1, segname2,...)</td>
<td>none</td>
<td>536</td>
</tr>
<tr>
<td>SEQERROR</td>
<td>(ABEND)</td>
<td>(ACCEPT)</td>
<td>(ABEND)</td>
<td>537</td>
</tr>
<tr>
<td>SORTWORK</td>
<td>(n,ccc,unitName, dataclas, mgmtclas, storclas)</td>
<td>(3,100,SYSDA)</td>
<td>545</td>
<td>Dynamically allocate sort work data sets.</td>
</tr>
</tbody>
</table>
Table 32  Reload function keywords (part 4 of 4)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPILL</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>547</td>
<td>Specify how to store segments while evaluating the SEQERROR parm.</td>
</tr>
<tr>
<td>TRUNC</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>556</td>
<td>Start each HIDAM or PHIDAM root in a new block.</td>
</tr>
<tr>
<td>UCBITMAP</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>557</td>
<td>Specify a separate value for up to ten data set groups.</td>
</tr>
<tr>
<td>USEREXIT</td>
<td>(exitName,, YES</td>
<td>none</td>
<td>566</td>
<td>Invoke a user exit routine.</td>
</tr>
<tr>
<td>USERHDR</td>
<td>See page 568</td>
<td>none</td>
<td>568</td>
<td>Create output records with a user-defined header (or prefix) to segment data.</td>
</tr>
</tbody>
</table>

Global options modules

You can specify options for the Reload function at the levels shown in Table 33. Because the Reload function uses the LOADPLUS global option modules, not all of the Reload function keywords are available in a global options module.

**NOTE**

You can override any option that is specified in an options module by including that option in the PLUSIN control statement data set.

Table 33  Ways to specify options for Reload function (part 1 of 2)

<table>
<thead>
<tr>
<th>Module</th>
<th>Used for</th>
<th>Resides in</th>
<th>Created by</th>
</tr>
</thead>
<tbody>
<tr>
<td>internal defaults</td>
<td>all jobs that don’t specify an options module or options in PLUSIN</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>LDPPARMS</td>
<td>all jobs at your installation</td>
<td>load library</td>
<td>ISPF panels or #DBGULBL batch job</td>
</tr>
<tr>
<td>LDP@imsi</td>
<td>all jobs run on a particular IMS system (for example, a test system)</td>
<td>load library</td>
<td>ISPF panels or #DBGULBL batch job</td>
</tr>
<tr>
<td>DFLT member of PDX</td>
<td>all jobs, all jobs run on a particular IMS system, or all jobs for a particular database, depending on how the options module was created</td>
<td>PDX</td>
<td>an earlier release of the product</td>
</tr>
</tbody>
</table>
You can combine the global options module, the PDX options member, and command methods for specifying options for maximum efficiency. You can use the LDPPARMS options module to specify the values you want to use for most of the databases at your site. For particular keywords you want to use only on a single IMS system (for example, a test system), you can use the LDP@imsi options module. For databases that need values different from those you specified in the LDPPARMS or LDP@imsi modules, you can create a PDX options member for those DBDs. For jobs that require values different from those you usually use for the database, you can specify the values in the PLUSIN data set.

You can specify values for the following options in a global options module for use with the Reload function:

- COMPRESS
- FMTRAA
- HDAMFSPF
- PDX
- PDXPARMS
- REPORTS
- SCAN
- SEQERROR
- TRUNC
- USEREXIT

If you specify options that the Reload function does not use, those options are ignored.

### Table 33  Ways to specify options for Reload function (part 2 of 2)

<table>
<thead>
<tr>
<th>Module</th>
<th>Used for</th>
<th>Resides in</th>
<th>Created by</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbdname member of PDX</td>
<td>all jobs for a particular database; PDX is not supported for HALDBs</td>
<td>PDX member</td>
<td>ISPF panels</td>
</tr>
<tr>
<td>PLUSIN</td>
<td>the job where the options are specified</td>
<td>JCL</td>
<td>adding keywords to PLUSIN control statement data set in reload job</td>
</tr>
</tbody>
</table>

You can create and modify the LDPPARMS and LDP@imsi global options modules by using the LOADPLUS ISPF interface. You can also create the DBD-level global members that are stored in the PDX. Figure 50 on page 221 shows the hierarchy of the Reload function ISPF interface.
The ISPF interface makes it easy for you to specify options at the IMSID level, DBD level, or both. Enter the required information on one of the global options panels. During execution, the utility retrieves settings from the global options module. Each IMSID and DBD can have a set of options associated with it.

Create and modify options as follows:

- To create a set of options for a particular DBD, select option 2 from the Reload Function primary menu.

- To create or modify a global options module (LDPPARMS or LDP@imsi), select option 3 from the Reload Function primary menu.

To scroll through the options panels, press **Enter**. When you press **Enter**, the utility validates the options you specified on the current panel and displays the next panel. The scroll keys (UP and DOWN) are not active. To go to a particular panel, enter its page number in the **Next Page** field of the current panel and press **Enter**.

When you make changes on these panels, you must press **Enter** before the utility validates your changes.
An explicit save is performed when you type `SAVE` on the Command line of a panel where SAVE is displayed as a valid command. If you do not want to save your changes, type `CANCEL` on the Command line and press Enter.

When you are finished, perform an implicit save by typing `END` on the Command line or pressing the function key designated as your END key (usually F3).

---

**NOTE**

Any keyword specified in the PLUSIN control statement data set overrides the corresponding keyword in the global options module.

---

Online help is available for each panel in the ISPF interface.

---

### Create using batch job

You can create and modify the LDPPARMS and LDP@imsi global options modules by executing the two LDPGLBLx steps in #DBUGLBL member of the sample library.

### Reports

The Reload function generates the reports shown in Table 34. For more information about the reports, refer to the page number shown.

#### Table 34  Reload function reports (part 1 of 2)

<table>
<thead>
<tr>
<th>Report</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset Statistics</td>
<td>591</td>
</tr>
<tr>
<td>DBD Summary</td>
<td>592</td>
</tr>
<tr>
<td>Disassembled DBD</td>
<td>594</td>
</tr>
<tr>
<td>Distribution of Database Record Sizes</td>
<td>594</td>
</tr>
<tr>
<td>Distribution of Free Space</td>
<td>595</td>
</tr>
<tr>
<td>Distribution of HDAM RAP Chain Lengths</td>
<td>595</td>
</tr>
<tr>
<td>HDAM Root Addressable Area Statistics</td>
<td>597</td>
</tr>
<tr>
<td>HDAM Root Placement Relative to Its Home Block</td>
<td>598</td>
</tr>
<tr>
<td>HD Sort - Records Count Report</td>
<td>599</td>
</tr>
<tr>
<td>Largest Database Records</td>
<td>600</td>
</tr>
<tr>
<td>Logical Relationship Information</td>
<td>601</td>
</tr>
<tr>
<td>Message Log</td>
<td>604</td>
</tr>
<tr>
<td>Options in Effect</td>
<td>604</td>
</tr>
</tbody>
</table>
HD Reorganization Reload differences

The Reload function differs from the IMS HD Reorganization Reload utility in the following ways:

- The Reload function formats the entire HDAM RAA rather than only to the highest numbered block to which a root randomizes. The FMTRAA keyword allows you to make the Reload function function similar to IMS. For more information, see “FMTRAA” on page 439.

- If you increase the size of a fixed-length segment between the unload and the reload of a database by using the BYTES operand in the DBD, the Reload function initializes the additional bytes with binary zeros. IMS fills the additional bytes with the values residing in the I/O area.

- The Reload function ignores the SCAN parameter in the DBD and uses a user-controlled space search algorithm instead:

  — When the randomized block for an HDAM or PHDAM database is full, IMS uses its space search algorithm to find space in the RAA. The Reload function has options that control how the search is performed. For more information, see “SCAN—space search method for HDAM and PHDAM” on page 531.

  — When a block becomes full for a HIDAM or PHIDAM database, IMS uses its space search algorithm to find space in a previous block before going to the next sequential block. The Reload function provides an option to control whether the search for space is in a previous block before going to the next sequential block. For more information, see “SCAN—space search method for HIDAM and PHIDAM” on page 531.

- For HDAM and PHDAM databases, the free block frequency factor specified in the DBD is not honored in the RAA. When a root randomizes to a block, the Reload function tries to place the root and its dependents into that block.

### Table 34  Reload function reports (part 2 of 2)

<table>
<thead>
<tr>
<th>Report</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent/Child Segment Statistics</td>
<td>606</td>
</tr>
<tr>
<td>Segment I/O Requirements Matrix</td>
<td>612</td>
</tr>
<tr>
<td>Segment Statistics</td>
<td>612</td>
</tr>
<tr>
<td>Status of Segments Read from HD Unload Data Set</td>
<td>613</td>
</tr>
<tr>
<td>Twin Segment Statistics</td>
<td>614</td>
</tr>
<tr>
<td>User Exit Routine Action</td>
<td>614</td>
</tr>
<tr>
<td>Variable Length Segment Statistics</td>
<td>615</td>
</tr>
</tbody>
</table>
For HDAM, PHDAM, HIDAM, and PHIDAM databases, IMS always places a segment in the same block in which its twin resides, if space is available. The Reload function does not always place the segment in the same block with its twin. This placement difference can occur when a segment has a large number of dependents that force the Reload function to fill the current block plus any number of subsequent blocks. The Reload function must place the next twin occurrence in the current block, not the block where the previous twin was placed. The IMS HD Reorganization Reload utility goes back to the block containing the previous twin. When the Reload function loads databases that have a wide range of segment sizes and long twin chains, you might see a greater probability of I/O from twin to twin during subsequent database operations.

If an HDAM or PHDAM database does not define a unique sequence field on an initial load or on an HD reload, IMS uses the insert rules of FIRST, LAST, or HERE to determine the sequence in which the roots are chained. The reload of an HDAM or PHDAM database reverses the order of the unsequenced roots when HERE or FIRST is used with the IMS reload.

The Reload function does not reverse the order of the unsequenced roots in the initial load or the reload. The Reload function chains the unsequenced segments in the same order as the input file.

The IMS HD Reorganization Reload utility does not issue a NOTIFY.REORG to DBRC if the reload step completion code is greater than 4. The Reload function always issues a NOTIFY.REORG, regardless of the completion code, if the database is registered to DBRC and the job step does not abend.

Performance considerations

You can affect the performance of the Reload function in several ways. This section discusses how to tune the Reload function for better performance.

Changing the HDAM or PHDAM randomizer

If you change the HDAM or PHDAM randomizing parameters (randomizer, RAPs, or RAA size), the physical sequence of the root segments might change. This could be true for any randomizers you created.

If the physical sequence of the root segments changes, the database must be sorted before it can be reloaded. If you attempt to reload a database where the records are not sorted and HDSORT(NO) is specified, the job terminates abnormally and an error message is issued. You must use the HD Sort utility or RAP Sequencer included with
the Reload function to sort the records. HDSORT(AUTO) is the internal default. HDSORT(AUTO) determines the optimal sort method to use—the HD Sort utility or the RAP Sequencer. To force a standard HD Sort, specify HDSORT(YES). For more information, see “HD Sort utility” on page 227.

Options

The Reload function options shown in Table 35 can affect the performance of the reload job and the database after the reload.

Table 35  Reload function options that affect performance

<table>
<thead>
<tr>
<th>Option</th>
<th>Reload performance</th>
<th>Database performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPRESS(YES)*</td>
<td>degrades</td>
<td>improves</td>
</tr>
<tr>
<td>FMTRAA(YES)*</td>
<td>degrades</td>
<td>improves</td>
</tr>
<tr>
<td>HDAMFSPF(IGNORE)</td>
<td>improves</td>
<td>improves</td>
</tr>
<tr>
<td>SCAN(n)</td>
<td>improves</td>
<td>improves</td>
</tr>
<tr>
<td>TRUNC(YES)</td>
<td>degrades</td>
<td>improves</td>
</tr>
<tr>
<td>USEREXIT(value)</td>
<td>degrades</td>
<td>no effect</td>
</tr>
</tbody>
</table>

*indicates default value

Multivolume data sets

The method that you use to allocate multivolume data sets differs for OSAM and VSAM.

VSAM data sets

Specify the VOLSERs in IDCAMS parameters.

OSAM data sets

OSAM data sets can cause problems because the volume table of contents (VTOC) handling is done on the last volume only. DS1LSTAR (the last block pointer) is valid on the last volume only.
Empty database initialization

If you reuse existing allocations, the last block pointer information might be wrong. To avoid using the wrong last block pointer information, BMC recommends that you delete and reallocate OSAM data sets.

If you are not using SMS for allocation, you can allocate OSAM data sets by using either of the following methods:

- Use one DD statement that specifies multiple VOLSERS and DISP=(,CATLG).
- Use multiple DD statements that specify DISP=(,KEEP), and use IDCAMS to catalog those data sets in a subsequent step.

Empty database initialization

Sometimes you need an empty database, particularly in an online environment. Because IMS does not allow the creation of a database online, the database must be created offline. Creating an empty database offline involves inserting one root segment and deleting it. This process initializes the database so it can be used online.

You can execute the RELOAD command to perform this initialization function with a JCL statement modification. Specify DFSUINPT DD DUMMY to create a usable empty database. A load PSB is not necessary.

You can initialize an empty database for the following database organizations and access methods:

- If the database is an HDAM or PHDAM database and you specify FMTRAA(YES), the entire RAA is formatted. If you specify FMTRAA(NO), only the first track is formatted. Formatting improves online performance when inserting segments into the database. For more information, see “FMTRAA” on page 439.
- If the database is a HIDAM or PHIDAM database, it contains a root segment with a key of all X’FF’s.
- If the database is a HISAM or SHISAM VSAM database, the KSDS is empty, but the high-used RBA is non-zero and the entry-sequenced data set (ESDS) contains a valid record 0.
- If the database is a SHISAM database, the KSDS is empty and the high-used RBA is non-zero.
HD Sort utility

The standard HD Sort utility, available in the Reload function, sorts an HD Unload data set into HDAM, PHDAM, HIDAM, PHIDAM, or HISAM order. The HD Sort utility is a functional alternative to the IMS System Utilities/Database Tools HDAM Physical Sequence Sort/Reload utility (PSSR). The HD Sort utility does not completely replace PSSR. This section provides an overview of the HD Sort utility.

You must invoke the HD Sort utility (before or during the RELOAD job step) when you perform any of the following tasks:

- change the key of a root segment
- convert an HDAM database to HIDAM or HISAM
- convert a PHDAM database to PHIDAM
- convert a HIDAM or HISAM database to HDAM
- convert a PHIDAM database to PHDAM
- change the randomizer parameters for an HDAM or PHDAM database (except maximum insert bytes)

To model proposed changes to an HDAM or PHDAM randomizer, use the HDAM Randomizer Analysis utility of POINTER CHECKER PLUS.

PSSR is designed to prevent cascading when you reload a database with the IMS Reorganization Reload utility. The HD Sort utility is designed to take advantage of the Reload function’s technique for preventing cascading (the SCAN option).

Figure 51 on page 228 shows where the HD Sort utility step fits within a simplified database reorganization process. Before and after the reorganization, take an image copy of the database and check the pointers. If the reorganized database has logical relationships or secondary indexes, you must run jobs to process these elements.

The HD Sort utility step normally runs after the DBDGEN that creates the new DBD and before the reload step. The Reload function performs the sort during the reload step when you include the HDSORT(YES) keyword on the RELOAD command. If you specify HDSORT(AUTO) on the RELOAD command, the Reload function determines whether the HD Sort utility or the RAP Sequencer is needed. The Reload function uses the method that is most efficient for the database. If you require the standard HD Sort utility, specify HDSORT(YES); otherwise, you can run the HD Sort utility as a stand-alone utility and specify HDSORT(NO) at reload time.

**NOTE**

For HALDBs, the HD Sort stand-alone utility does not support override of randomizer parameters.
The HD Sort utility sorts an HD Unload data set. The HD Sort utility does not run in a full IMS environment; it uses only the information included in the DBD. Therefore, if you are using a customized randomizer that expects register 7 to contain the process scheduling table (PST) address, register 7 will not be initialized because it requires a full IMS environment. When the randomizing routine gets control, registers 0 and 7 point to dummy blocks instead of IMS control blocks. The HD Sort utility uses registers 1, 9, 13, 14, and 15 in the same way IMS uses them.

If VLSHRT(YES) is the DFSORT installation default specification, you must include a DFSPARM DD statement to override this default. For details, see “JCL requirements” on page 229.

**NOTE**

The HD Sort utility sorts only database records using the root key. It does not sort dependents. If the key values or lengths are changed for dependents, the HD Sort utility does not resequence the dependents.
**System flow**

Figure 52 shows the system flow of the HD Sort utility. The HD Sort utility invokes your installation’s sort/merge utility to sort the HD Unload-format records into physical sequence. Inputs are the HD Unload data set, the DBD (the new one if you are changing it), the HDAM or PHDAM randomizing routine, and the PLUSIN control statement data set. Outputs are the sorted HD Unload data set and the BMCMMSG reports.

**JCL requirements**

Figure 53 shows generic JCL for executing the HD Sort utility.

---

**Figure 53  HD Sort utility JCL (part 1 of 2)**

```plaintext
//HDSORT JOB (acct),'userName',MSGCLASS=m,NOTIFY=user,CLASS=c
//SORT EXEC PGM=HDSORT
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMS.RESLIB
//DFSRESLIB DD DISP=SHR,DSN=IMS.RESLIB
//IMS DD DISP=SHR,DSN=ims.dbdlib.new
//RECON1 DD DISP=SHR,DSN=your.recon1 ===for HALDB only; required
//RECON2 DD DISP=SHR,DSN=your.recon2 ===for HALDB only; required
//RECON3 DD DISP=SHR,DSN=your.recon3 ===for HALDB only; required
//SORTIN DD DISP=SHR,DSN=database.hdunld.unsorted,DISP=SHR
//SORTOUT DD DSN=database.hdunld.sorted,DISP=(NEW,CATLG),UNIT=TAPE
```
An explanation of the JCL follows:

**EXEC**

Required. The EXEC statement must specify PGM=HDSORT. The HD Sort utility does *not* run under the IMS region controller (DFSRRC00).

**STEPLIB DD**

Required. Include the following libraries in the concatenation:

- the BMC load library
- library containing the HDAM randomizer

**NOTE**

Because the sort product does not search the LNKLST concatenation, this statement is required even if the BMC load library is included in the LNKLST concatenation.

**IMS DD**

Required. Describes the IMSVS.DBDLIB data set that contains the DBD that will be used to sort the unsorted unload file into sequence.

**RECONx DD**

Required for HALDBs. Defines the DBRC RECON data sets.
SORTIN DD

Required. Defines the input data set that contains the HD Unload-format records that you want the HD Sort utility to sort. The HD Unload data set can be created by the following utilities:

- Unload function. The HD Sort utility supports the abbreviated records that the Unload function creates with the DFSURGU1(LONG), DFSURGU1(SHORT), DFSURGU1(XSHORT), and USERHDR options. It also supports the compressed records that the Unload function creates with the EXPAND(NO) option. If the HD Unload data set was created by the Unload function but the header records were deleted by the user, the records must be in DFSURGU1(LONG) format for the HD Sort utility to sort them correctly. If the header record has been deleted and the USERHDR keyword is not specified with the SORT command, the HD Sort utility assumes that the records are in DFSURGU1(LONG) format.

- IMS HD Reorganization Unload utility

- High Speed Sequential Retrieval (HSSR) feature of the IMS System Utilities/Database Tools package

- any other utility that creates HD Unload-format records

SORTOUT DD

Required. Defines the output data set to contain the sorted HD Unload-format records. You can use this data set as input to the following utilities:

- the Reload function
- the IMS HD Reorganization Reload utility
- any other utility that reads HD Unload-format records

DFSPARM DD

Required when you use the HD Sort utility with DFSORT and VLSHRT=YES is the DFSORT default. Specify the following statement:

```
//DFSPARM DD *
OPTION NOVLSHRT
```

BMCPRINT DD

Optional. For more information, see “BMCPRINT DD” on page 354.
SORTWKnn DD

Optional. If you specify the SORTWORK option, the Reload function dynamically allocates the sort work data set. Defines the sort work areas that your site’s sort/merge utility uses to sort the HD Unload-format records. For more information, see the documentation for your site’s sort/merge utility.

SYSOUT DD

Required. Defines the print data set to contain messages from the sort/merge utility. It is normally a SYSOUT-type data set.

PLUSIN DD

Required. Defines the data set that contains the control statements for the HD Sort utility. You must include a control statement in the PLUSIN data set to identify the DBD name of the database to sort. You can also include keywords to override the existing HDAM or PHDAM randomizer parameters. These keywords allow you to run the DBDGEN after the HD Sort utility step; however, you normally should run the DBDGEN before the HD Sort utility step.

**NOTE**
The SORT command is required.

Table 36 describes the valid keywords.

**Table 36  PLUSIN DD keywords (part 1 of 2)**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>Required. Specify the DBD name of the database. This name normally refers to the new DBD. There is no default value. If you enter a different DBD name on the control statement than the DBD that was used to create the HD Unload data set, the HD Sort utility uses the randomizing parameters from the DBD that is named in the control statement.</td>
</tr>
<tr>
<td>DFSDBUXI</td>
<td>Optional. Specify whether to use the DFSDBUXI user exit. For more information, see “DFSDBUXI” on page 424.</td>
</tr>
<tr>
<td>DESCICODE</td>
<td>Optional. Specify a descriptor code. For more information, see “MONITOR, MONUSERS, ROUTCODE, DESCICODE” on page 490.</td>
</tr>
<tr>
<td>HDPRINT</td>
<td>Optional. Specify an output data set. For more information, see “xxxPRINT” on page 522.</td>
</tr>
<tr>
<td>KEYLEN</td>
<td>Optional. Specify a value (1-3 digits) for the length (in bytes) of the root segment’s key field. Valid values range from 1 to 255 (inclusive). Leading zeros are not required. The default value is the key length that is specified in the DBD.</td>
</tr>
</tbody>
</table>
Example

The following example shows the sorting of an HD Unload data set that was created from a HIDAM database. The DBD has been changed to define an HDAM database (HDAMDB). The new DBD defines the use of the DFSHDC10 randomizer, a maximum RBN of 40000, 3 RAPs per block, and a 32-byte root segment key field.

```
//PLUSIN DD *
SORT -
DBD(HDAMDB) RMNAME(DFSHDC10) -
NBLOCKS(40000) -
NRAPS(3) -
KEYLEN(32)
```
In the Reload function, the API lets you initially load an IMS database much faster than DL/I can load the database, and the API provides meaningful statistics. The API uses the Reload function to construct the database. Database segments are presented to the Reload function in hierarchical sequence from a user-written load program rather than from the HD Unload data set. DL/I calls using the AIB interface are supported. The API does not access the PDX.

**NOTE**
The API is *not* a complete replacement for full-function DL/I.

**System flow**

Figure 54 shows the system flow of the API.
The API works as follows:

1. Because the API uses standard DL/I batch JCL, the IMS region controller (DFSRRC00) is invoked by the MVS Initiator, which initializes the region for batch DL/I. BMC provides a module with an alias of DFSPR000. When the BMC-provided DFSPR000 module receives control for each DL/I call, it analyzes and intercepts DL/I calls to the PCBs you designated it to handle (through the PCB and/or PCBNAME keyword). All other DL/I calls to other PCBs are passed to the IBM-provided DFSPR000 module.

2. When the application program makes a DL/I call, the BMC-provided DFSPR000 module receives control.

3. The BMC-provided DFSPR000 module checks the PCB:
   - If the PCB is to be processed by the Reload function, control passes to the BMC product.
   - If the PCB is not to be processed by the Reload function, control passes to the IBM-provided DFSPR000 module and the call is serviced by DL/I.

JCL requirements

To allow a DL/I batch program to use the API, add several DD statements to the batch program’s run-time JCL. The PSB must use a PCB that has a PROCOPT of L or LS. The application program uses the standard DL/I call interface. You do not need to recompile or relink the application program.

Figure 55 shows sample JCL to load a database using a DL/I user load program with the API.

Figure 55 Sample JCL to run a user load program using the API (part 1 of 2)

```plaintext
//USERLOAD EXEC PGM=DFSRRC00,PARM="DLI,loadpgm,loadpsb"
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=your.program.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
// DD DISP=SHR,DSN=IMSVS.PSBLIB
//RECON1 DD DISP=SHR,DSN=your.recon1 ENDfor HALDB only; required
//RECON2 DD DISP=SHR,DSN=your.recon2 ENDfor HALDB only; required
//RECON3 DD DISP=SHR,DSN=your.recon3 ENDfor HALDB only; required
//DFSVSAMPE DD DISP=SHR,DSN=IMSVS.PROCLIB(VSAM01)
//IEFRDER DD DISP=OLD,DSN=iefrder.log
//BMCPRINT DD SYSOUT=*```

JCL requirements
Use the following statements in your user load program’s JCL. If the database contains logical relationships or secondary indexes, the DFSURWF1 DD statement is required.

**NOTE**
For OSAM databases, if the DD statement describing the database data set contains the DISP=(NEW,KEEP,) parameter, you must also specify a VOL=SER=xxxxxxxx parameter.

**EXEC**

Specify PGM=DFSRRC00,PARM='DLI,loadpgm,loadpsb', where loadpgm is the name of the application program and loadpsb is the PSB name. You can use DBB rather than DLI in the PARM operand.

**STEPLIB DD**

Required. Must point to the BMC load library. If several DD statements appear in the STEPLIB statement, this statement must be the first in the concatenation. The STEPLIB must be ahead of the RESLIB.

**NOTE**
All libraries in the STEPLIB concatenation must be APF authorized, or you must have started the DBUSS.

**DFSRESLB DD**

Required. Describes the library that contains the IMS RESLIB.

**IMS DD**

Required. Must point to the DBD library and the PSB library that contain the DBDs and the PSBs for the PCBs that are being serviced by the Reload function.
**RECONx DD**

Required for HALDBs. Defines the DBRC RECON data sets.

**IMSACB DD**

Required only if the first PARM operand in the EXEC statement specified DBB rather than DLI. Describes the library containing the ACB for the PSB referenced in the PARM operand of the EXEC statement.

**APITRACE DD**

Optional. Contains output from the TRACE command. The APITRACE data set is dynamically allocated if it is needed.

**PLUSIN DD**

Required. Contains the commands and keywords for the API to use. The USERLOAD command is required. Valid keywords are described with the USERLOAD command.

**USERLOAD command**

The USERLOAD command allows you to initially load a database faster than you can with the IMS utilities. The USERLOAD command is required. The following keywords are optional:

- PCB
- PCBNAME
- PRINT

You can also specify all of the keywords that are valid with the RELOAD command in the PLUSIN data set with the USERLOAD command, with the following exceptions:

- CKUPDATE (any value)
- COMPRESS (NO)
- DYNALLOC (any value)
- IDCAMS (any value)
- PART (any value)
- PDX (OKAYnnnn) or (STOPnnnn)
- SEQERROR (ACCEPT) or (ACCEPT,nn)
- SEGPOS (any value)
The PDX keyword is ignored. If the PDX data set is not available, the user load continues with a return code of zero. The other option values cause a user 2716 abend. The default values for the options are acceptable.

### Table 37 USERLOAD command keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDSORT(AUTO)</td>
<td>If you specify HDSORT(AUTO) or accept it as the default, the following values are used:</td>
</tr>
<tr>
<td></td>
<td>- For HDAM and PHDAM databases, HDSORT(YES) is used.</td>
</tr>
<tr>
<td></td>
<td>- In all other cases, HDSORT(NO) is used.</td>
</tr>
<tr>
<td>PCB</td>
<td>The PCB keyword allows you to specify one or more PCBs within the PSB that the API processes. You can specify the PCB by number or DBD name. When you specify by name, the API processes the first matching occurrence of the DBD. A maximum of 255 characters is allowed for this keyword.</td>
</tr>
<tr>
<td>PCBNAME</td>
<td>The PCBNAME keyword provides an alternate method of specifying one or more PCBs within the PSB that the API processes. You can specify the PCB by the PCB name assigned to it in the label field or PCBNAME keyword of the PCB statement. A maximum of 255 characters is allowed for this keyword.</td>
</tr>
<tr>
<td>PRINT</td>
<td>The PRINT keyword prints the DL/I call, the PCB information, and the contents of the user I/O area. If you want all information (CALL, PCB, and IOAREA), specify PRINT(ALL). The information prints to the APITRACE DD statement. If you omit the PRINT keyword, the default value is ALL.</td>
</tr>
</tbody>
</table>

**Example 1:**
The following example shows how to specify keywords for the USERLOAD command when you use the PCB keyword:

```
USERLOAD PCB(dbdName|pcbNumber) + PRINT(ALL|CALL.PCB.IOAREA)
```

**Example 2:**
The following example shows how to specify keywords for the USERLOAD command when you use the PCBNAME keyword:

```
USERLOAD PCBNAME(pcbname) + PRINT(ALL|CALL.PCB.IOAREA)
```
Example 3:
In the following example, the USERLOAD command tells the API to service all DL/I calls to the second PCB in the PSB. (The PSB is specified in the EXEC statement’s PARM.) This example also generates a printed trace of the call, PCB contents, and I/O area.

```
USERLOAD PCB(2) PRINT(ALL)
```

DL/I status codes

The API generates status codes for each DL/I ISRT call that are similar to IMS status codes. Table 38 lists the DL/I status codes and their meanings.

Table 38  Reload function API DL/I status codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>AM</td>
<td>Only ISRT calls are supported.</td>
</tr>
<tr>
<td>AH</td>
<td>Too few parameters were detected in the call.</td>
</tr>
<tr>
<td>AC</td>
<td>An invalid segment name was encountered in the SSA.</td>
</tr>
<tr>
<td>AI</td>
<td>The program was unable to open a data set.</td>
</tr>
<tr>
<td>AJ</td>
<td>The load program’s PCB contains an unsupported command code.</td>
</tr>
<tr>
<td>AO</td>
<td>An I/O error occurred.</td>
</tr>
</tbody>
</table>
| LB   | A segment’s sequence field already exists for the parent.  
     | or A segment type does not have a PTF pointer, and the segment being inserted is  
     | either the second segment of this segment type for the parent or the second HDAM  
     | or PHDAM root for one anchor point. |
| LC   | The key field of the segment is out of sequence. |
| LD   | No parent was loaded for this segment. |
| LE   | The sequence of sibling segments is not the same as the sequence in the DBD. |
| $1   | Too many parameters were detected in the call. |
| $2   | Only one PCB can be used per job step. |

Restrictions

User load programs operate under the following limitations:

- Only ISRT calls are processed by the Reload function-serviced PCB to the database that is being loaded.
- All ISRT calls must use the same PCB.
- You can load only one database per job step.

- Instead of passing a PCB status code back to the caller, the API might issue an abend.

- You cannot use the ddname BMCMSG for a user data set unless the DCB attributes include RECFM=FB or FBA. The BMCMSG data set is shared with the Reload function.

- If an HDAM or PHDAM database does not define a unique sequence field and the insert rules are FIRST or HERE, IMS chains the duplicate segments in reverse order (LAST IN, FIRST OUT).

  The API chains the segments in the same order as they are inserted (FIRST IN, FIRST OUT).

- The API does not support a PSB with field-level sensitivity.

- Do not move the BMC-provided DFSPR000 module to your IMS RESLIB. This will overlay the IMS-provided module, which can cause problems for other applications.

- The only acceptable command code is D (path calls). If the PCB contains an unsupported command code, the API returns an AJ status code to the load program.

- Command code V can be coded, but it is ignored.

- HDAM and PHDAM load programs must specify PROCOPT=L. HIDAM and PHIDAM load programs must specify PROCOPT=LS.

- The Reload function supports HDAM and PHDAM database loads in physical block sequence only. If an HDAM or PHDAM load program cannot present root segments to the API in physical (randomized block number) order, you must specify HDSORT(YES) in the PLUSIN control statement data set.

- When you specify HDSORT(YES), a blank status code indicates that the segment was passed to the sort. The actual insert of the segment into the database could subsequently fail with a user abend.

- Does not support batch pipes.
WF1 Split function

If you do not use the WF1 Split function, the DFSURWF1 data set contains all secondary index and logical relationship records. Processing this data set can take time. To speed the reorganization process, you can separate the secondary index records from the logical relationship records. The logical relationship records are written to PRPURWF1. The secondary index records are written to individual files for parallel sort processing.

You can use the WF1 Split function in either of the following ways:

- Include a DFSURWF1 DD statement and an HLDURIDX DD statement in the Reload function job-step JCL. The data set that is specified in the DFSURWF1 DD statement is used for logical relationship processing. The data set specified by the HLDURIDX DD statement is used for secondary index processing. These data sets are not interchangeable.

- Specify the OUTDDX keyword in the Reload function job and the INDDX keyword in the Index Build function keyword.

Some reasons for using this function follow:

- By splitting the two sets of WF1-type records, you can process the logical relationship records and the secondary index records concurrently. Concurrent processing shortens the total load or reorganization cycle. Each index can be written to a separate file, so the indexes can be created concurrently.

- If the amount of data in the WF1 data set exceeds the DASD sort work space, you might be forced to use tape for sorting. When the one large data set is split into two smaller data sets, the sort product requires less DASD sort work space.

- The WF1 Split function generates a record count of the logical relationship and secondary index records. This count can be passed to a prefix resolution utility by using the PARM='FILSZ=n' parameter. Providing the sort product with the input file size improves sort performance.

The data sets containing the secondary index records are passed to the BUILD function. The data set containing the logical relationship records is passed to the Prefix Resolution/Update function or the IMS Prefix Resolution utility. In either case, the prefix resolution process runs significantly faster and uses less DASD sort work space.

To use the WF1 Split function, specify the following keywords:

- the Reload function OUTDDX keyword—place secondary index work file records in data sets other than DFSURWF1
the Index Build function INDDX keyword—read secondary index records from a data set other than DFSURWF1

Use the HLDURIDX DD statement to split logical relationship and secondary index records without creating a separate data set for each index. If you want to split the secondary index records from the logical relationship records, specify a DFSURWF1 DD statement and an HLDURIDX DD statement.

The HLDURIDX data set is optional; if used, it contains all secondary index records. The DFSURWF1 data set contains logical relationship records. The HLDURIDX data set can be used as input to Index Build function if you specify the INDD keyword.
Index Build function

This chapter describes how to run the Index Build function as a stand-alone utility. This chapter includes the following topics:

Overview .................................................. 244
JCL requirements ........................................... 245
  Build secondary indexes only .......................... 245
  Build secondary indexes and primary index in one job step .......... 246
  Build primary index only ............................... 247
  DD statements ............................................ 247
  PLUSIN examples ........................................ 248
Sort considerations ........................................ 249
Keywords .................................................... 249
Global options modules .................................... 251
  Create using ISPF interface ............................ 252
  Create using batch job .................................. 253
Reports ....................................................... 253
Simplified index creation and maintenance ......................... 254
Other functions ............................................. 257
  Load indexes from an existing DFSURWF1 file .................. 257
  Build indexes by scanning the primary database .................. 257
  Build primary indexes .................................... 258
  HALDBs ...................................................... 259
  Create a new secondary index ............................ 260
  Rebuild an existing secondary index ........................ 261
  Maintain a secondary index .............................. 261
  Create an empty secondary index .......................... 262
Overview

The Index Build function (invoked with the BUILD command) loads any of the following indexes:

- all secondary indexes of a database
- selected secondary indexes
- the primary index of a HIDAM or PHIDAM database
- the ILDS of a PHDAM or PHIDAM database

Using the PRIMIND keyword, you can build the primary and secondary indexes in one step (and one scan of the database).

The Index Build function makes it easy to create and maintain JCL. You can use existing SECONDARY INDEX UTILITY or MAXM Reorg classic JCL. The Index Build function automatically converts most SECONDARY INDEX UTILITY and MAXM Reorg classic keywords to the Index Build function format.

When you implement a new secondary index for a primary database, add to the JCL a DD statement that defines the new secondary index.

The Index Build function can scan the primary database, or it can use the input data set that you specify with the INDD keyword. BMC recommends that you use the INDD(DFSURWF1) keyword if the Index Build function is being executed during a complete database reorganization. Using an INDD keyword can eliminate redundant processing to recreate index records that might already exist in a DFSURWF1 file. If you do not specify an INDD keyword, the Index Build function scans the primary database to create the index records.

The Index Build function works as follows:

1. During initialization, the Index Build function obtains information about the secondary indexes from the DBD of the primary (target) database.

2. If no INDD keyword is specified, the Index Build function scans the primary database and creates the index records. The Index Build function selects the scan method (hierarchical or sequential) that is most efficient for the database. If the job specifies a SCAN keyword (supported in earlier releases), processing continues but the SCAN keyword is ignored.

3. The Index Build function invokes your installation’s sort product to sort the index records by index and VSAM key field.

4. The Index Build function loads the selected indexes.
Figure 56 shows the system flow for the Index Build function.

**Figure 56  Index Build function system flow**

The Index Build function accepts output from the following utilities:

- Reload function
- IMS HD Reorganization Reload

The Index Build function provides an automated interface to DBRC. For more information, see Appendix C, ”DBRC support.”

**JCL requirements**

This section discusses the JCL requirements for the Index Build function.

**Build secondary indexes only**

Figure 57 shows the JCL requirements for building secondary indexes when scanning the database. BMC recommends that you use this method when you do not have existing WF1 records, such as those created during a reorganization.

**Figure 57  JCL to build secondary indexes only (part 1 of 2)**

```plaintext
//SCANLOAD EXEC PGM=BMCSIU
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
// IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
// or
// DBDLIB DD DISP=SHR,DSN=IMSVS.DBDLIB
// RECON1 DD DISP=SHR,DSN=your.recon1 <==for HALDB only; required
```
Build secondary indexes and primary index in one job step

Figure 58 shows the JCL requirements for building the primary index and the secondary indexes in the same job step. BMC recommends that you use this method when you need to build the primary index and all secondary indexes.

This JCL builds the primary index and all secondary indexes by scanning the primary database (no INDD is specified). The Index Build function attempts to perform a sequential scan for the primary index and any secondary indexes that meet the requirements for a sequential scan. If a sequential scan is not possible for secondary indexes, the Index Build function performs a hierarchical scan in the same job step. The Index Build function builds the indexes using as few passes through the database as possible.

For examples of keywords used in the PLUSIN control statement data set, see “PLUSIN examples” on page 248.
Figure 59 shows the JCL requirements for building the primary index only.

**Figure 59  JCL to build primary index**

```plaintext
//SCANLOAD EXEC PGM=BMCSIU
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLIB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//* or
//DBDLIB DD DISP=SHR,DSN=IMSVS.DBDLIB
//RECON1 DD DISP=SHR,DSN=your.recon1 <==for HALDB only; required
//RECON2 DD DISP=SHR,DSN=your.recon2 <==for HALDB only; required
//RECON3 DD DISP=SHR,DSN=your.recon3 <==for HALDB only; required
//primedb DD DISP=OLD,DSN=primary.database
//primeind DD DISP=OLD,DSN=primary.index
//BMCMMSG DD SYSOUT=*  
//BMCPRINT DD SYSOUT=*  
//PLUSIN DD *
BUILD DBD(PRIMEDB) PRIMIND(ONLY)
```

**DD statements**

For more information about the DD statements, see Chapter 13, “JCL requirements.” The following DD statements are specific to the Index Build function BUILD command:

**primedb DD**

Optional; however, if you specify DYNALLOC(YES) and the data set is registered to DBRC, the Index Build function dynamically allocates this data set. Do not include this DD statement if you specify DYNALLOC(YES). This DD statement describes the data set that is used for the primary DBD. The ddname must match the name in the primary database DBD.

**secindex DD**

Optional; however, if you specify DYNALLOC(YES) and the data set is registered to DBRC, the Index Build function dynamically allocates this data set. Do not include this DD statement if you specify DYNALLOC(YES). This DD statement describes the VSAM data set that contains the secondary index database. The ddname must match the name in the secondary index’s DBD.
primeind DD

Optional; however, if you specify DYNALLOC(YES) and the data set is registered to DBRC, the Index Build function dynamically allocates this data set. Do not include this DD statement if you specify DYNALLOC(YES). This DD statement describes the data set that is used for the primary index. The ddname must match the name in the primary index DBD.

PLUSIN examples

In the following examples, the primary database is named CUSTOMER.

Example 1:
The primary and all secondary indexes are built in the same step:

```plaintext
BUILD DBD(CUSTOMER) PRIMIND(YES)
```

Example 2:
Only the primary index is built in this step:

```plaintext
BUILD DBD(CUSTOMER) PRIMIND(ONLY)
```

Example 3:
The secondary indexes named SEC1 and SEC2 are built in the same step as the primary index:

```plaintext
BUILD DBD(CUSTOMER) SECINDEX(SEC1,SEC2) PRIMIND(YES)
or
BUILD DBD(CUSTOMER) PRIMIND(ONLY)
BUILD DBD(CUSTOMER) SECINDEX(SEC1)
BUILD DBD(CUSTOMER) SECINDEX(SEC2)
```

Example 4:
The Index Build function dynamically allocates four sort work files per secondary index task (maximum of 3) and 150 cylinders for each data set on the unit named UNIT1:

```plaintext
BUILD DBD(CUSTOMER) SORTWORK(4,150,UNIT1)
```

Example 5:
The secondary indexes of database CUSTOMER are built from work file 1 records:

```plaintext
BUILD DBD(CUSTOMER) INDD(DFSURWF1)
```
Sort considerations

Allocation of the sort work data sets plays a large part in the efficiency of the Sort utility. The Index Build function provides the following methods for determining the size of these data sets:

1. The Index Build function passes estimated record counts to the Sort utility from the values provided in the FILSZ keyword (“FILSZ” on page 437).

2. If you do not supply a FILSZ value for an index and use the IDCAMS keyword, the Index Build function extracts the records counts from the ICF Catalog before performing the IDCAMS DELETE/DEFINE of the index. These counts are then passed to the Sort utility. For more information, see “IDCAMS” on page 464.

3. If the record counts are not available from 1 or 2 above, the Index Build uses the value from the SORTWORK keyword (“SORTWORK” on page 545).

4. If none of the methods listed above are available, the Index Build function uses the values from the installation default module (DLIGSET0).

If method 1 or 2 is used, the Sort utility allocates appropriately sized sort work data sets and is able to optimize its use of memory. If method 3 or 4 is used, the Index Build function allocates the same size work files for all sorts and the Sort utility uses system defined default values for its memory allocation. For more information, see “SIUSORT” on page 542.

Keywords

The Index Build function uses the BUILD command. Table 39 lists valid keywords.

Table 39  Index Build function keywords (part 1 of 3)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>(dbname1,</td>
<td>none</td>
<td>420</td>
<td>Specify the DBD name for the primary DBD.</td>
</tr>
<tr>
<td></td>
<td>dbname2,...)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBRC</td>
<td>(YES)</td>
<td>Use the DBRC specification in the IMSCTRL macro.</td>
<td>422</td>
<td>Specify whether to invoke DBRC. For HALDBs, DBRC(YES) is required.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DESC柘CDE</td>
<td>(nn,...)</td>
<td>(7)</td>
<td>490</td>
<td>Specify descriptor codes used to route messages to the console.</td>
</tr>
<tr>
<td></td>
<td>max 10 entries</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>(YES)</td>
<td>(NO)</td>
<td>430</td>
<td>Dynamically allocate database data sets using data set names specified in the RECON data sets. For HALDBs, DYNALLOC(YES) is required; it is also the default.</td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Keyword</td>
<td>Accepted values</td>
<td>Default values</td>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------</td>
<td>----------------</td>
<td>------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FILSZ</td>
<td>(INDEX) (NO)</td>
<td>INDEX</td>
<td>437</td>
<td>Specify an estimated number of records for each index file. The value will be passed to DFSORT to determine the input file size.</td>
</tr>
<tr>
<td>GSGNAME</td>
<td>(gsgName)</td>
<td>Use the GSG name used in system generation.</td>
<td>442</td>
<td>Use a global service group (GSG) name for remote site recovery (RSR).</td>
</tr>
<tr>
<td>ICNEEDED</td>
<td>(ON) (OFF)</td>
<td>(ON) for RECOV databases (OFF) for NONRECOV databases</td>
<td>457</td>
<td>Specify whether to turn off the ICNEEDED flag for all NONRECOV index DBDs. This is affected by the value of the NONRECOV keyword in the DLIGSET0 module.</td>
</tr>
<tr>
<td>IDCAMS</td>
<td>(input)</td>
<td>none</td>
<td>464</td>
<td>Specify SYSIN and SYSPRINT files for IDCAMS processing.</td>
</tr>
<tr>
<td>ILDS</td>
<td>(NO) (ONLY) (YES)</td>
<td>(NO)</td>
<td>468</td>
<td>Specify to build or initialize Indirect List Data Sets.</td>
</tr>
<tr>
<td>IMSID</td>
<td>(imsID)</td>
<td>Use the IMSID from IMS generation (DFSVC000 module).</td>
<td>470</td>
<td>Specify IMSID under which reports are stored in the PDX.</td>
</tr>
<tr>
<td>INDD</td>
<td>(DFSURWF1)</td>
<td>none</td>
<td>471</td>
<td>Specify the input data set if you are not scanning the primary database.</td>
</tr>
<tr>
<td>INDDX</td>
<td>(ddname)</td>
<td>none</td>
<td>472</td>
<td>Specify the data set to read secondary index records from.</td>
</tr>
<tr>
<td>MONITOR</td>
<td>(nnnnnnnn)</td>
<td>no automatic monitoring</td>
<td>490</td>
<td>Monitor job progress.</td>
</tr>
<tr>
<td>MONUSERS</td>
<td>(user01, user02,...)</td>
<td>Send monitor messages to TSO users and BMCMSG.</td>
<td>490</td>
<td>Send monitor messages to up to 10 specified TSO user IDs.</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>(YES) (NO)</td>
<td>(YES)</td>
<td>504</td>
<td>Enable parallel processing.</td>
</tr>
<tr>
<td>PART</td>
<td>(ALL) (partitionName,...) (nn,nn,...)</td>
<td>(ALL)</td>
<td>508</td>
<td>Perform reorganization tasks for one or more database partitions.</td>
</tr>
<tr>
<td>PDX</td>
<td>(OKAYnmmm) (STOPnmmm)</td>
<td>(OKAY0004)</td>
<td>518</td>
<td>Specify action to take if the PDX fails to open. PDX is not supported for HALDBs.</td>
</tr>
<tr>
<td>PRIMIND</td>
<td>(YES) (NO) (ONLY)</td>
<td>(NO)</td>
<td>520</td>
<td>Specify to build primary indexes.</td>
</tr>
</tbody>
</table>
You can specify options for the Index Build function at the levels shown in Table 40. Because the Index Build function uses the SECONDARY INDEX UTILITY global option modules, not all Index Build function keywords are available in a global options module.

NOTE
You can override any option specified in an options module by including that option in the PLUSIN control statement data set.

Table 40  Ways to specify options for the Index Build function

<table>
<thead>
<tr>
<th>Module</th>
<th>Used for</th>
<th>Resides in</th>
<th>Created by</th>
</tr>
</thead>
<tbody>
<tr>
<td>internal defaults</td>
<td>all jobs that don’t specify an options module or options in PLUSIN</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>SIUPARMS</td>
<td>all jobs at your installation</td>
<td>load library</td>
<td>ISPF panels or #DBUGLBL batch job</td>
</tr>
<tr>
<td>SIU@imsi</td>
<td>all jobs run on a particular IMS system (for example, a test system)</td>
<td>load library</td>
<td>ISPF panels or #DBUGLBL batch job</td>
</tr>
<tr>
<td>PLUSIN</td>
<td>the job where the options are specified</td>
<td>JCL</td>
<td>adding keywords to PLUSIN control statement data set in reload job</td>
</tr>
</tbody>
</table>
You can combine the global options module and command methods for specifying options for maximum efficiency. You can use the SIUPARMS options module to specify the values that you want to use for most of the databases at your site. For particular keywords that you want to use only on a single IMS system (for example, a test system), you can use the SIU@imsi options module. For jobs that require values different from those you usually use for the database, you can specify the values in the PLUSIN data set.

You can specify values for the following options in a global options module for use with the Index Build function:

- DBRC
- ICNEEDED
- PDX

The Index Build function ignores those options that it does not use.

Create using ISPF interface

You can create and modify the global options modules by using the Index Build Function ISPF interface. Figure 60 shows the hierarchy of the interface.
The ISPF interface makes it easy for you to specify options at the IMSID level. Enter the required information on one of the global options panels. During execution, the utility retrieves settings from the global options module. Each IMSID can have a set of options associated with it.

To create or modify a global options module (SIUPARMS or SIU@imsi), select option 2 from the Index Build Function primary menu.

To scroll through the options panels, press Enter. When you press Enter, the utility validates the options you specified on the current panel and displays the next panel. The scroll keys (UP and DOWN) are not active. To go to a particular panel, enter its page number in the Next Page field of the current panel and press Enter.

The utility does not validate your changes until you press Enter.

An explicit save is performed when you type SAVE on the Command line of a panel where SAVE is displayed as a valid command. If you do not want to save your changes, type CANCEL on the Command line and press Enter.

When you are finished, perform an implicit save by typing END on the Command line or pressing the function key designated as your END key (usually F3).

**Create using batch job**

You can create and modify the SIUPARMS and SIU@imsi global options modules by executing the two SIUGLBLx steps in the #DBGULBL member of the sample library.

**Reports**

The Index Build function generates the reports shown in Table 41.

<table>
<thead>
<tr>
<th>Report</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datasets Used</td>
<td>592</td>
</tr>
<tr>
<td>Index Load Output Results</td>
<td>600</td>
</tr>
<tr>
<td>Message Log</td>
<td>604</td>
</tr>
<tr>
<td>Options in Effect</td>
<td>604</td>
</tr>
<tr>
<td>Secondary Index Definition</td>
<td>609</td>
</tr>
</tbody>
</table>
Simplified index creation and maintenance

The Index Build function replaces any or all of the IMS utilities that are required for secondary index processing.

The Index Build function reduces the personnel time required to create and test JCL and the CPU time required to build a secondary index because the Index Build function reduces the multi-step IMS procedure (see Figure 61) to a one-step procedure (see Figure 62 on page 255).

Figure 61  Secondary index creation with the IMS utilities
When you use the standard IMS utilities, creating a secondary index requires six steps:

1. Execute the IMS Prereorganization utility to create control information required by the subsequent IMS utilities.

2. Execute the HD Unload utility to unload the database.

3. Execute the HD Reload utility to load the database.

4. Execute the Prefix Resolution utility to sort the index records.

5. Execute the HISAM Unload utility to convert sorted index records to the format required by the HISAM Reload utility.

6. Execute the HISAM Reload utility to load the index.

Steps 5 and 6 are repeated for multiple secondary indexes.

The Index Build function simplifies this process to a one-step process. Just supply commands in the PLUSIN data set and execute the Index Build function. Figure 62 shows this process.

**Figure 62  Secondary index creation with the Index Build function**
Figure 63 shows how flexible the Index Build function is in replacing IMS utilities. If the primary database contains logical relationships, you must also run a prefix resolution utility.

**Prereorganization:**
If there are no logical relationships, the IMS Prereorganization utility is not needed.

**Prefix Resolution, HISAM Unload, and HISAM Reload:**
The Index Build function reads the output from IMS HD Reorganization Reload utility (the DFSURWF1 file) and writes the secondary index databases.

Figure 63  Index Build function (no logical relationships)
Other functions

This section discusses other functions that are available with the Index Build function.

Load indexes from an existing DFSURWF1 file

When you execute the Index Build function during a database reorganization, it is most efficient to use the DFSURWF1 records from the load or reload, rather than scanning the primary database again; for example:

```
BUILD DBD(dbdName) INDD(DFSURWF1)
```

Build indexes by scanning the primary database

If you do not specify an INDD keyword, the Index Build function scans the primary database to create index work records. The Index Build function uses one of the following scanning methods.

**Sequential scan**

A sequential scan reads blocks sequentially rather than hierarchically. The Index Build function reads blocks, identifies segments, and extracts the data necessary to build an index. Because the Index Build function is not reading the database in hierarchical order, the scan can complete significantly faster than a traditional hierarchical scan.

A sequential scan has the following requirements:

- If any concatenated key information (such as /CK) is used in fields (such as SUBSEQ, DDATA), the source segment must be the root segment.

- If the source segment is not the target segment, the target must be the parent of the source and the source must have physical parent (PP) pointers to the target.

- For HALDBs, the root segment must be the target segment.

- If the index has symbolic pointers, the source segment must be the root segment.
Hierarchical scan

A hierarchical scan reads each segment in a database in hierarchical order. This is the traditional scanning method.

The Index Build function chooses the best scanning method for your database. The SCAN keyword is ignored. The Index Build function attempts to perform a sequential scan for the primary index and any secondary indexes that meet the requirements for a sequential scan. If a sequential scan is not possible for secondary indexes, the Index Build function performs a hierarchical scan in the same job step. The Index Build function builds the indexes using as few passes through the database as possible.

Build primary indexes

Figure 64 shows JCL for building a HIDAM or PHIDAM primary index.

**Figure 64  JCL for building a primary index**

```
//LOAD EXEC PGM=BMCSIU
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
// IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
// RECON1 DD DISP=SHR,DSN=IMSVS.RECON1
// RECON2 DD DISP=SHR,DSN=IMSVS.RECON2
// BMCMSG DD SYSOUT=* 
// BMCPRT DD SYSOUT=* 
// PLUSR IN DD *
   BUILD DBD(dbdName) PRIMIND(ONLY) IDCAMS(file1, file2) -
       DYNALLOC(YES) DBRC(YES)
/*
//file1 DD DISP=DSN=your.idcmsg.control.statements 
//file2 DD SYSOUT=* 
```

PRIMIND(ONLY) tells the Index Build function to build only the primary index. For more information, see “PRIMIND” on page 520.

The IDCAMS keyword specifies file names. The first file is a sequential input file that contains the IDCAMS control statements to delete and define the data set. The second file is an optional output file; it contains the SYSOUT of the IDCAMS execution. If you do not specify an output file, the Index Build function uses the BMCMSG data set.

Specify DYNALLOC(YES) to dynamically allocate the primary database and index data sets using names specified in the RECON data sets. Dynamic allocation requires DBRC. Specify DBRC(YES), or ensure that the use of DBRC was specified in the IMSCTRL macro during the IMS system generation process.
HALDBs

To build a subset of index partitions for a HALDB, use one of the following methods.

**NOTE**
The Index Build function might also build a subset of index partitions when you run an unload and reload of a subset of a HALDB’s main database partitions. When the key ranges of the main database partitions use the same source segment offset and length as the partitioned secondary index, the Index Build function builds only the affected partitions.

### Primary indexes and ILDSs

To build primary indexes and Indirect List Data Sets (ILDSs) for selected HALDB partitions, use the PART parameter.

The following rules apply to building primary indexes and ILDSs for selected HALDB partitions.

- In the partition list, specify the main database partitions for which you want to build either primary indexes or ILDSs.
- For primary indexes, specify PRIMIND(ONLY).
- For ILDSs, specify ILDS(ONLY).
- You cannot specify both the PART and SECINDEX parameters.
- You cannot specify the following combinations:
  - ILDS(YES) and PRIMIND(ONLY)
  - ILDS(ONLY) and PRIMIND(YES)

The following JCL shows an example that results in primary index builds for partitions CUST2A and CUST3A.

```jcl
BUILD DBD(CUSTOMER) PRIMIND(ONLY) PART(CUST2A,CUST3A)
```

The following JCL shows an example that results in ILDS for partitions CUST2A and CUST3A.

```jcl
BUILD DBD(CUSTOMER) ILDS(ONLY) PART(CUST2A,CUST3A)
```
**Secondary indexes**

To build a subset of secondary index partitions, specify the SECINDEX parameter with a list of the secondary index partitions that you want to build.

**NOTE**

The Index Build function generally reads the entire database in this case.

The following rules apply to building a subset of secondary index partitions:

- In the partition list, specify the secondary index partition names that you want to build.
- You cannot specify both the PART and SECINDEX parameters.

**Create a new secondary index**

Figure 65 shows the system flow of the Index Build function when you are creating a new secondary index for an existing database. The BUILD command executes the scan and load functions in the same OS job step.

**Figure 65 Creating a secondary index**

You can use the Index Build function to create a new secondary index if either of the following situations is true:

- The target segment in the primary database is the root segment.
- The target segment in the primary database already has a secondary index.
If the target segment in the primary database is not a root segment and does not have
an existing secondary index, you must create the physical parent pointers. You can
create the physical parent pointers in two ways:

- Run the Reorg function.
- Create the index using traditional methods and utilities.

To create a new secondary index using traditional methods, you must create index
records before building the index. Follow these steps:

1. Run an unload utility such as the Unload function.
2. Define a new index DBD.
3. Define the index in the primary database (using XDFLD).
4. Run a reload utility such as the Reload function to create DFSURWF1 records.
5. Run an index utility such as the Index Build function using the DFSURWF1 records
   as input.

Rebuild an existing secondary index

The Index Build function can recreate or rebuild a secondary index without
unloading and reloading the primary database. Instead of recovering the secondary
index from an image copy, execute the Index Build function. You can eliminate
regular image copies of secondary indexes without compromising recoverability. You
can also take advantage of the NORECOV option in DBRC.

Maintain a secondary index

When you reorganize a primary database that has one or more secondary indexes,
you must rebuild the secondary indexes. Use the BUILD command with the
INDD(DFSURWF1) keyword to build the secondary index after the primary database
is reloaded with a reload utility. For example

```
BUILD DBD('primedb') INDD(DFSURWF1)
```

The output of the reload utility serves as input to the Index Build function. Because
this index data is not sequenced, the Index Build function invokes your site’s sort
product.
Create an empty secondary index

You can create an empty secondary index by using the BUILD command and a DD DUMMY input JCL statement. For an example, see Figure 66.

**Figure 66  JCL for creating an empty secondary index**

```
//LOAD EXEC PGM=BMCSIU
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// IMS DD DISP=SHR,DSN=IMSVS.RESLIB
// RECON1 DD DISP=SHR,DSN=your.recon1  <===for HALDB only; required
// RECON2 DD DISP=SHR,DSN=your.recon2  <===for HALDB only; required
// RECON3 DD DISP=SHR,DSN=your.recon3  <===for HALDB only; required
// BMCM5G DD SYSOUT=*  
// BMCPRTY DD SYSOUT=* 
// SECXDB2 DD DISP=OLD,DSN=SECONDARY.INDEX.DATABASE2 
// DFSURWF1 DD DUMMY 
// PLUSIN DD * 
BUILD DBD(PRIMEDBD) INDD(DFSURWF1) SECINDEX(SECXDB2)
/*
```

An empty VSAM KSDS does not contain any logical records; the high-used RBA is not zero. Execute the BUILD command after you create an empty primary database by using the Reload function, the IMS Fast Reload utility, or any other DL/I load program to create valid empty secondary indexes.
Copy function

This chapter describes the Copy function and healing EPS pointers. This chapter contains the following topics:

Overview .......................................................... 263
Healing EPS pointers in a HALDB ................................ 268
    POINTER CHECKER PLUS .................................. 268
    Healing EPS pointers with the Copy function .......... 269

Overview

You can use the Copy function online or offline to heal EPS pointers in a HALDB. The Copy function reorganizes indexes without reorganizing the primary database. It is useful for removing CI/CA splits. It is similar to a REPRO but uses shadow databases while the database is online for updates; Copy takes advantage of with the AOI and swapping automation that the Online Reorg function provides. You can also use the Copy function to produce copies of databases for use in other systems by specifying SWAP(N). The following features are supported by the COPY command:

- The HEAL keyword heals Extended Pointer Set (EPS) pointers.

**NOTE**

Running under the COPY command, this keyword replaces the HEALEPS function in earlier releases.

- SHARE keyword support was added to support online or offline COPY.
- With a CHANGE RECORDING FACILITY the COPY command uses the CHANGE RECORDING FACILITY log capture facility to allow copying of database data sets while they are online.
- IDCAMS(MODEL) support was added.
You can perform the following operations with the COPY command:

— create a copy of a production database, both full function and HALDB (the whole database or one or more partitions including the ILDS)
— reorganize indexes and remove CI/CA splits
— move data sets to new volumes
— heal HALDB pointers in secondary indexes and logical relationships
— copy a subset of partitions of HALDB databases and indexes
— copy HALDB ILDS and/or primary index

**NOTE**

To use the COPY command, you must meet the following requirements:

- You must have a license for either FAST REORG FACILITY/EP or CHANGE RECORDING FACILITY.
- You must have the CRF product active in the IMS control region.
- The COPY job must use the same RECONs as the IMS control region.
- Multiple COPY commands require Cross-System Image Manager (XIM) as each COPY is run in a separate XIM discrete unit of work (UOW).

The following example shows sample JCL to execute the Copy function.

```plaintext
//***********************************************************
//* SAMPLE JCL
//***********************************************************
//CPYTSTA EXEC PGM=DLICPYON,REGION=4096K
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
//DD DISP=SHR,DSN=IMSVS.Rxx.RESLIB
//DFSRESLBDD DISP=SHR,DSN=IMSVS.Rxx.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.Rxx.DBMLIB
//PLUSIN DD *
COPY DBD(dbdName) SWAP(N) -
DBRC(Y) IMSID(xxxx) DYNALLOC(Y,Y,0)
//SYSPRINT DD SYSOUT=* 
//BMCPRINT DD SYSOUT=* 
//BMCMMSG DD SYSOUT=* 
//BMCTRACE DD SYSOUT=* 
```

The following JCL statements apply to the Copy function:

**EXEC**

Required. Specifies PGM=DLICPYON.
STELIB DD

Required. Points to the BMC load library.

IMS DD

Required. Defines the IMSVS.DBDLIB data set that contains the DBD that will be used to sort the unsorted unload file into sequence.

DFSRESLB DD

Required. Defines the library that contains the IMS.RESLIB.

dbddnam DD

Required unless DYNALLOC is used. Defines the new copy of the database after it is copied.

newddnam DD

Required unless DYNALLOC is used. Defines the data set that contains the original database that is being reorganized.

PLUSIN DD

Required. Defines the data set that contains the control statements for the Copy function.

The COPY command is required.

Table 42 describes the keywords that you use with PLUSIN DD.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Required or optional</th>
<th>Accepted values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOIUSER</td>
<td>optional</td>
<td>(progName)</td>
<td>specifies whether to invoke a user exit from within the AOI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>default: none</td>
<td></td>
</tr>
<tr>
<td>DBD</td>
<td>required</td>
<td>(dbddName)</td>
<td>specifies the DBD name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>default: none</td>
<td></td>
</tr>
</tbody>
</table>

If you are reorganizing a single index, specify the name of the INDEX DBD. If you are making an offline copy of a database, specify the name of the database that is being copied. The name can be a secondary index, a primary index, or a primary DBD. If you specify a primary database name, you can build multiple indexes at once by specifying the SECINDEX keyword.
### Table 42  Copy function JCL PLUSIN DD keywords (part 2 of 3)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Required or optional</th>
<th>Accepted values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBRC</td>
<td>required</td>
<td>(Y) (N)</td>
<td>specifies whether to invoke DBRC when there is no region controller. Specify N to invoke only by region controller.</td>
</tr>
<tr>
<td>DESCNAME</td>
<td>optional</td>
<td>(nn,...)</td>
<td>maximum ten entries default: (7) specifies descriptor codes used to route messages to the console</td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>optional</td>
<td>(YES) (NO) (YES,YES,x) (YES,NO,x) (NO,YES) default: (NO) dynamically allocate all database data sets using the data set names specified in the RECON data sets, and give output data sets unique names</td>
<td></td>
</tr>
<tr>
<td>GSGNAME</td>
<td>optional</td>
<td>(gsgName) default: Use the GSG name used in system generation uses a global service group (GSG) name for remote site recovery (RSR)</td>
<td></td>
</tr>
<tr>
<td>HEAL</td>
<td>optional</td>
<td>(Y) (N) default: (N) heals or corrects Extended Pointer Set (EPS) pointers in HALDBs</td>
<td></td>
</tr>
<tr>
<td>IDCAMS</td>
<td>optional</td>
<td>(input) (input,output) (input,output,Y) (input,output,N) (model) default: none specifies SYSIN and SYSPRINT files for IDCAMS processing DBRC(Y) required.</td>
<td></td>
</tr>
<tr>
<td>ILDS</td>
<td></td>
<td>(YES) (NO) (ONLY) default: (NO) valid for PHDAM and PHIDAM specifies whether the ILDS of a HALDB should be copied</td>
<td></td>
</tr>
<tr>
<td>INDD</td>
<td>optional</td>
<td>(DFSURWF1) (DFSURIDX) default: none specifies the input data set if you are not scanning the primary database</td>
<td></td>
</tr>
</tbody>
</table>
### Table 42  Copy function JCL PLUSIN DD keywords (part 3 of 3)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Required or optional</th>
<th>Accepted values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTDD</td>
<td>optional</td>
<td>(ddname)</td>
<td>specifies the DD statement for the copied database</td>
</tr>
<tr>
<td>PART</td>
<td></td>
<td>(partName1, partName2...)</td>
<td>specifies the names of partitions that are to be copied</td>
</tr>
<tr>
<td>PRIMIND</td>
<td></td>
<td>(YES) (NO) (ONLY)</td>
<td>specifies whether the primary index of a HALDB should be copied</td>
</tr>
<tr>
<td>ROUTCODE</td>
<td>optional</td>
<td>(nn,...)</td>
<td>specifies routing codes to control the destination of messages that are sent to the console</td>
</tr>
</tbody>
</table>
| SECINDEX| optional             | (YES) (NO) (ONLY) | ■ YES–copy the DBD and all secondary indexes  
          |                      | ■ NO–copy only the DBD  
          |                      | ■ ONLY–copy only the secondary indexes of the DBD  |
| SHARE   | optional             | (YES,UPDATE)    | specify whether users will have update access to the database during reorganization  
          |                      | online  
          | optional             | (YES) (NO) (YES,READ) | specifies whether users will have read access to the database during reorganization  
          |                      | (DELETE) (YES,IC) (SUSPEND,IC) | (YES) (NO) (YES,READ) | specifies whether users will have read access to the database during reorganization  
          |                      | default: (NO)      | specifies whether to swap the data set names after a successful reorganization |
Healing EPS pointers in a HALDB

HALDBs use Indirect List Datasets (ILDSs) when logical relationships or secondary indexes exist in the database design. The pointers between relationships can change as a partition is reorganized. The ILDS is used to locate the relocated data after reorganization. These pointers are corrected by IMS as the data is accessed with an updated PSB and are then considered healed.

MAXM Reorg provides a method (using the Copy function) for mass healing of the pointers that are stored in the ILDS. Each index or relationship can be healed individually or in mass while updates are allowed (under MAXM Reorg/Online for IMS) or in exclusive mode (using MAXM Reorg/EP for IMS). Either approach uses the AOI function to control online access during the copy.

NOTE

If you have Cross-System Image Manager (XIM) installed, you can specify multiple COPY commands in the PLUSIN stream. If multiple COPY commands are found, XIM spawns an address space for each COPY command. If you do not use XIM, and you specify multiple COPY commands, the step fails.

POINTER CHECKER PLUS

In addition to healing pointers with the Copy function, you can use the POINTER CHECKER PLUS product to report unhealed pointers. Use the CHKILDS keyword while running a FULL check of all databases.

In the following example, the control card runs reports on a PHIDAM database with three partitions and six PSINDEX databases, some with multiple partitions:

```bash
//PCPSYSIN DD *
GLBL TYPE(FULL) IMSID(IMSA) DYNDDB(DB) REPORTS(ALL) DBRC(Y)
CHECKALL DBD(XIUOBD) PART(ALL) CHK(ALL) CHKILDS(Y)
```
The following example shows reports from POINTER CHECKER PLUS after a reorganization of one partition:

<table>
<thead>
<tr>
<th>R U N S U M M A R Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS/ESA 9.1.0 DATE=2010.236 08/24/2010 TIME=12:10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DBDNAME</th>
<th>DBORG</th>
<th>PARTNAME</th>
<th>DDNAME</th>
<th>DSG</th>
<th>METHOD</th>
<th>COUNT</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>XIU0DBD</td>
<td>PHIDAM</td>
<td>XIU0A</td>
<td>XIU0AL</td>
<td>KSDS</td>
<td>NO</td>
<td>367</td>
<td>VERIFY OKAY</td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>PHIDAM</td>
<td>XIU0A</td>
<td>XIU0AX</td>
<td>OSAM</td>
<td>1,080</td>
<td>VERIFY OKAY</td>
<td></td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>PHIDAM</td>
<td>XIU0B</td>
<td>XIU0BL</td>
<td>KSDS</td>
<td>VERIFY OKAY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>PHIDAM</td>
<td>XIU0B</td>
<td>XIU0BX</td>
<td>KSDS</td>
<td>562</td>
<td>VERIFY OKAY</td>
<td></td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>PHIDAM</td>
<td>XIU0B</td>
<td>XIU0BA</td>
<td>OSAM</td>
<td>2,403</td>
<td>VERIFY OKAY</td>
<td></td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>PHIDAM</td>
<td>XIU0C</td>
<td>XIU0CL</td>
<td>KSDS</td>
<td>NO</td>
<td>190</td>
<td>VERIFY OKAY</td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>PHIDAM</td>
<td>XIU0C</td>
<td>XIU0CX</td>
<td>KSDS</td>
<td>1,620</td>
<td>VERIFY OKAY</td>
<td></td>
</tr>
</tbody>
</table>

HALDB HEALED/UNHEALED COUNTS
FROM DBD XIU0NDX1 PART XIU0X1A SEGMENT EAPTEST1

<table>
<thead>
<tr>
<th>TARGET</th>
<th>TARGET</th>
<th>TARGET</th>
<th>HIGHEST</th>
<th>HEALED</th>
<th>UNHEALED</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>SEGMENT</td>
<td>PART</td>
<td>REORG</td>
<td>POINTER</td>
<td>POINTER</td>
</tr>
<tr>
<td>NAME</td>
<td>NAME</td>
<td>NAME</td>
<td>NUMBER</td>
<td>COUNT</td>
<td>COUNT</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>---------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>BA#AASEG</td>
<td>XIU0A</td>
<td>1</td>
<td>38</td>
<td>0</td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>BA#AASEG</td>
<td>XIU0B</td>
<td>2</td>
<td>0</td>
<td>62</td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>BA#AASEG</td>
<td>XIU0C</td>
<td>3</td>
<td>17</td>
<td>0</td>
</tr>
</tbody>
</table>

HALDB HEALED/UNHEALED COUNTS
FROM DBD XIU0NDX2 PART XIU0X2A SEGMENT EAPTEST2

<table>
<thead>
<tr>
<th>TARGET</th>
<th>TARGET</th>
<th>TARGET</th>
<th>HIGHEST</th>
<th>HEALED</th>
<th>UNHEALED</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>SEGMENT</td>
<td>PART</td>
<td>REORG</td>
<td>POINTER</td>
<td>POINTER</td>
</tr>
<tr>
<td>NAME</td>
<td>NAME</td>
<td>NAME</td>
<td>NUMBER</td>
<td>COUNT</td>
<td>COUNT</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>---------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>BA#AASEG</td>
<td>XIU0A</td>
<td>1</td>
<td>248</td>
<td>0</td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>BA#AASEG</td>
<td>XIU0B</td>
<td>2</td>
<td>0</td>
<td>367</td>
</tr>
<tr>
<td>XIU0DBD</td>
<td>BA#AASEG</td>
<td>XIU0C</td>
<td>3</td>
<td>150</td>
<td>0</td>
</tr>
</tbody>
</table>

Healing EPS pointers with the Copy function

The following example shows a control card that heals all pointers in all PSINDEX relationships while allowing online updates to the database:

```
//PLUSIN DD *
COPY DBD(XIU0DBD) SWAP(Y) DBRC(Y) IDCAMS(MODEL) -
   DYNALLOC(Y,Y,C) SHARE(Y,UPDATE) HEAL(Y) SECINDEX(ONLY)
```
The following example shows reports from the Copy function in BMCMSG:

```plaintext
BMC250634I DATABASE(XIU0NDX1) DD(XIU0X1AA) SEGMENT(EAPTEST1) SEGMENT COUNT(117) HEALED(62)
BMC180304I COPY DATABASE 'XIU0NDX1' ENDED. INDD(XIU0X1AA) OUTDD(SYS00033).
BMC250634I DATABASE(XIU0NDX4) DD(XIU0X4AA) SEGMENT(EAPTEST4) SEGMENT COUNT(765) HEALED(367)
BMC180304I COPY DATABASE 'XIU0NDX4' ENDED. INDD(XIU0X4AA) OUTDD(SYS00037).
BMC250634I DATABASE(XIU0NDX2) DD(XIU0X2AA) SEGMENT(EAPTEST2) SEGMENT COUNT(765) HEALED(367)
BMC180304I COPY DATABASE 'XIU0NDX2' ENDED. INDD(XIU0X2AA) OUTDD(SYS00034).
BMC250634I DATABASE(XIU0NDX5) DD(XIU0X5CA) SEGMENT(EAPTEST5) SEGMENT COUNT(1393) HEALED(742)
BMC180304I COPY DATABASE 'XIU0NDX5' ENDED. INDD(XIU0X5CA) OUTDD(SYS00040).
BMC250634I DATABASE(XIU0NDX3) DD(XIU0X3BA) SEGMENT(EAPTEST3) SEGMENT COUNT(2660) HEALED(1215)
BMC180304I COPY DATABASE 'XIU0NDX3' ENDED. INDD(XIU0X3BA) OUTDD(SYS00036).
BMC250634I DATABASE(XIU0NDX5) DD(XIU0X5BA) SEGMENT(EAPTEST5) SEGMENT COUNT(5533) HEALED(2210)
BMC180304I COPY DATABASE 'XIU0NDX5' ENDED. INDD(XIU0X5BA) OUTDD(SYS00039).
BMC250634I DATABASE(XIU0NDX5) DD(XIU0X5AA) SEGMENT(EAPTEST5) SEGMENT COUNT(7416) HEALED(2761)
BMC180304I COPY DATABASE 'XIU0NDX5' ENDED. INDD(XIU0X5AA) OUTDD(SYS00038).
BMC250634I DATABASE(XIU0NDX3) DD(XIU0X3AA) SEGMENT(EAPTEST3) SEGMENT COUNT(11682) HEALED(4498)
BMC180304I COPY DATABASE 'XIU0NDX3' ENDED. INDD(XIU0X3AA) OUTDD(SYS00035).
BMC250634I DATABASE(XIU0NDX6) DD(XIU0X6AA) SEGMENT(EAPTEST6) SEGMENT COUNT(143742) HEALED(57533)
BMC180304I COPY DATABASE 'XIU0NDX6' ENDED. INDD(XIU0X6AA) OUTDD(SYS00041).
BMC180375I COMMAND COPY (1) ENDED
```

The following example heals an individual PSINDEX by naming the PSINDEX in the DBD keyword:

```plaintext
//PLUSIN    DD *
COPY DBD(XIU0NDX1) SWAP(Y) DBRC(Y) IDCAMS(MODEL) -
        DYNALLOC(Y,Y,C) SHARE(Y,UPDATE) HEAL(Y)
```
Working with logical relationships

This chapter describes how to run the Prefix Resolution and Prefix Update functions. This chapter contains the following topics:

Overview ................................................................. 272
High-performance Prefix Resolution function ........................................ 272
  JCL requirements ..................................................... 275
Concurrent Prefix Update function ................................................... 277
  JCL requirements ..................................................... 278
Commands and keywords ....................................................... 278
Global options modules ....................................................... 280
  Create using ISPF interface ........................................ 281
  Create using batch job ............................................. 283
Reports ................................................................. 284
Performance considerations .................................................... 285
Other prefix resolution and update functions ..................................... 286
  IMS-compatible Prefix Resolution function .................................. 286
  Two-step Prefix Resolution function .................................. 289
  Stand-alone Prefix Update function .................................. 295
Secondary Index Resolution function ............................................. 297
  JCL requirements ..................................................... 299
Work File 1 Generator ..................................................... 300
  JCL requirements ..................................................... 302
Work File Splitter .......................................................... 304
  JCL requirements ..................................................... 305
Overview

The Prefix Resolution/Update function enables you to resolve and update prefixes for logically related databases. You can use several different methods to resolve and update prefixes:

- IMS-compatible Prefix Resolution function
- High-performance Prefix Resolution function
- Two-step Prefix Resolution function
- Stand-alone Prefix Update function
- Concurrent Prefix Update function

BMC recommends that you use the High-performance Prefix Resolution function and the Concurrent Prefix Update function for best performance.

The Prefix Resolution/Update function also offers additional functions:

- Secondary Index Resolution function
- Work File 1 Generator
- Work File Splitter

High-performance Prefix Resolution function

The High-performance Prefix Resolution function operates much like the IMS-compatible Prefix Resolution function, except that it does not create the DFSURWF2 data set.

The High-performance Prefix Resolution function passes the output of the first sort directly to the second sort using an internal buffer. The High-performance Prefix Resolution function provides significant performance improvements over the IMS-compatible Prefix Resolution function, but it increases the amount of sort work space required to approximately the amount required for the IMS Prefix Resolution utility.

To run the High-performance Prefix Resolution function, you can use existing IMS Prefix Resolution utility JCL by making minor changes.

Figure 67 on page 273 shows the system flow of the High-performance Prefix Resolution function.
The High-performance Prefix Resolution function works as follows:

1. The High-performance Prefix Resolution function reads the DBD library to get information about the relevant logical relationships.

2. The High-performance Prefix Resolution function reads the control data set (DFSURCDS) that was generated by the IMS Prereorganization utility.
3. If DBRC is active, the High-performance Prefix Resolution function reads and writes recovery control information to the RECON data sets.

4. The DFSURWF1 or SORTIN data set supplies the logical relationship records and can supply the secondary index records that were created during the database load, reload, or scan.

The DFSURWF1 data set always contains IMS-format logical relationship and can also contain secondary index (type 40) records.

The SORTIN data set can contain either (but not both) of the following record types:

- BMC-format logical relationship records. The SORTIN data set cannot contain secondary index records when it contains BMC-format logical relationship records. Specify SORTIN(N) to indicate that the records are in BMC format.

- IMS-format logical relationship records and secondary index records. Specify SORTIN(Y) to indicate that the records are in IMS format.

5. The High-performance Prefix Resolution function (program name DFSURG10 or PRPURG10) consists of two sort phases, PRPURG20 and PRPURG30, that run concurrently. To run the phases concurrently, you must specify HIPO(Y). Both phases invoke your site’s sort utility.

- The first phase, Parent-Child Resolution (program name PRPURG20), sorts logical child relationship records with their corresponding logical parent records and resolves the logical child and logical parent pointers. This phase uses the SORTWKnn data sets for sorting. It passes the partially resolved work file records directly to the second sort phase, using an internal buffer.

- The second phase, Logical Twin Resolution (program name PRPURG30), sorts the records into physical database sequence (within the data set) and resolves any logical twin forward and backward pointers. This phase uses the SRT2WKnm data sets for sorting. This phase creates and sorts logical relationship work file (DFSURWF3) records, which you use as input to the Prefix Update function (PRPURGP0). You can run the Prefix Update function concurrently with PRPURG30 if you supply a DD statement defining the data set to be updated and specify UPDATE (Y).

6. The High-performance Prefix Resolution function extracts any index records from the DFSURWF1 or SORTIN data set during the input phase of PRPURG20.

- If you specify SORTIDX(Y), the High-performance Prefix Resolution function sorts the records by internally invoking the Secondary Index Resolution function (PRPURG40). It creates a temporary PRPURIDX work data set for use during the sort.
If you specify SORTIDX(N), you must process the index records by using the Secondary Index Resolution function (PRPURG40), the Index Build function, or the IMS Prefix Resolution utility.

7. The High-performance Prefix Resolution function captures and reports statistics. It writes these reports to the SYSPRINT data set, and it can write them to the PDX data set.

---

**JCL requirements**

Figure 68 shows the JCL requirements for the High-performance Prefix Resolution function.

**Figure 68  High-performance Prefix Resolution function JCL**

```
//JOBNAME   JOB , 'USER',CLASS=A,MSGCLASS=X,NOTIFY=USER
//******************************************************
//PREFIX     EXEC PGM=DFSURG10,REGION=0M
//STEPLIB    DD DISP=SHR,DSN=bmc.xxx.load
//           DD DISP=SHR,DSN=IMSVS.RESLIB
//SYSPRINT   DD SYSOUT=*  SY2PRINT DD SYSOUT=*  SYSPRINT DD SYSOUT=*
//SYSOUT     DD SYSOUT=*  SYSOUT DD SYSOUT=*  SY2OUT DD SYSOUT=*  SYLOUT DD SYSOUT=*  SYLOUT DD SYSOUT=*  SYSUDUMP DD SYSOUT=*  IMS DD DSN=IMSVS.DBDLIB,DISP=SHR
//SORTWK01   DD UNIT=SYSDA,SPACE=(CYL,(3,2))  SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(3,2))  SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(3,2))  SRT2WK01 DD UNIT=SYSDA,SPACE=(CYL,(2,1))  SRT2WK02 DD UNIT=SYSDA,SPACE=(CYL,(2,1))  SRT2WK03 DD UNIT=SYSDA,SPACE=(CYL,(2,1))  DFSURWF1 DD DISP=SHR,DSN=dfsurwf1.data.set  DFSURWF3 DD DISP=.PASS,DSN=dfsurwf3.data.set  DFSURCDS DD DISP=SHR,DSN=dfsurcds.data.set  PLUSIN DD *  GBL SORTIN(Y) HIPO(Y) UPDATE(Y) SORTDD(SRT2) - BUFSIZE(4096) NBUF(16) SIZE(65535)
```

---

**WARNING**

You do not need to provide DCB information about any output data set for any Prefix Resolution/Update function job. If you do not specify the block size, the High-performance Prefix Resolution function computes the optimum block size for the device. It uses half-track blocking for 3380 and 3390 devices and full-track blocking for other disk devices. Blocks on magnetic tape will be about 32 KB. Using the DCB recommendations as stated in the IMS Utilities Reference manual will degrade performance.
For more information about DD statements, see Chapter 13, “JCL requirements.” The following DD statements are specific to the High-performance Prefix Resolution function:

**EXEC**

Required. Must specify DFSURG10 or PRPURG10. Do not execute the High-performance Prefix Resolution function under the IMS region controller (PGM=DFSRRC00). BMC recommends a region size of 0M. The High-performance Prefix Resolution function does not accept the CORE parameter on this statement.

**physical database DD**

Optional. Defines the work data set that the High-performance Prefix Resolution function creates during the Logical Twin Resolution phase. This data set contains DFSURWF3-format logical relationship records for a physical database as identified by the physical database name. This DD statement tells the High-performance Prefix Resolution function to write all logical child and logical parent records for the physical database to this data set, unless you specify physical data set DD statements to split the records.

**physical data set DD**

Required if you specify UPDATE(Y); otherwise, optional.

- If you specify UPDATE(Y), this statement defines a database data set that the High-performance Prefix Resolution function updates using concurrent prefix update.

- If you specify UPDATE(N), this statement defines the work data set created during the Logical Twin Resolution phase. (This data set contains DFSURWF3-format logical relationship records for a physical data set as identified by the physical data set ddname, or DSGROUP). When you include this DD statement, the High-performance Prefix Resolution function writes all logical child and logical parent records for the physical data set to this data set.

**SORTWKnn DD**

Required. Defines the first set of sort work data sets.

---

**NOTE**

The High-performance Prefix Resolution function requires a sort program that can specify the four-byte prefix for all SORTxxxx data sets in the program-invoked sort calling sequence. All IBM DFSORT-compatible products should have this capability. The sort work resources required are double that of the other methods because two sorts are running concurrently. Each group of sort work data sets must be large enough to sort the entire logical input file; consult your site’s sort manual to determine the actual sizes.
Although the High-performance Prefix Resolution function requires two sets of sort work data sets, the total space requirements are approximately the same as for the IMS Prefix Resolution utility. Allocate 55 to 60 percent of the space to the SORTWK\text{nn} data sets and 40 to 45 percent to the SRT2WK\text{nn} data sets.

**SRT2WK\text{nn} DD**

Required. Defines the second set of sort work data sets.

**SY2PRINT DD**

Optional. Same as SYSPRINT for the second sort output.

**SRTOUT DD**

Optional. Defines the sort message data set for the second sort. If you do not provide this data set, the High-performance Prefix Resolution function dynamically allocates it to the job message class.

**Concurrent Prefix Update function**

The Prefix Update function (program name PRPURGP0) replaces the IMS utilities that update the pointers in the prefixes of logically related segments to establish the logical relationships. You can run the Prefix Update function as a stand-alone utility, under the IMS region controller, or concurrently as the output of any Prefix Resolution function.

The Prefix Update function (PRPURGP0) can be invoked during the Logical Twin Resolution phase (PRPURG30) of any Prefix Resolution function. Specify UPDATE(Y) with the HIPO function. When you specify UPDATE(Y), prefixes are updated for any database data set group that has a DD statement present in the JCL. All other output records are written to the DFSURWF3 data set.

You can run the Prefix Update function concurrently with the Logical Twin Resolution phase (program name PRPURG30/DFSURG10) by specifying UPDATE(Y). Concurrent Prefix Update updates any physical data set that has a DD statement for it in the JCL; the Logical Twin Resolution phase does not write work file records for those data sets to DFSURWF3. The Concurrent Prefix Update function improves performance about 10 percent over Stand-alone Prefix Update.
JCL requirements

Figure 69 shows the JCL requirements for the Concurrent Prefix Update function. To update prefixes concurrently with a Prefix Resolution function, use the JCL you use for the Prefix Resolution function and DD statements for the databases. Ensure that the PLUSIN control statement data set specifies UPDATE(Y).

Figure 69 Concurrent Prefix Update function JCL requirements

```
//JOBNAME JOB , 'USER', CLASS=A, MSGCLASS=X, NOTIFY=USER
//***********************************************************
//PREFIX EXEC PGM=DFSURG10, REGION=0M
//STEPLIB DD DISP=SHR, DSN=bmc.xxx.load
// DD DISP=SHR, DSN=IMSVS.RESLIB
//SYSPRINT DD SYSOUT=*  //SYSOUT DD SYSOUT=*  //PLUSOUT DD SYSOUT=*  //PLUSLIST DD SYSOUT=*  //SYSUDUMP DD SYSOUT=*  //IMS DD DSN=IMSVS.DBDLIB,DISP=SHR
//SORTWK01 DD UNIT=SYSDA, SPACE=(CYL,(5,3))  //SORTWK02 DD UNIT=SYSDA, SPACE=(CYL,(5,3))
//SORTWK03 DD UNIT=SYSDA, SPACE=(CYL,(5,3))  //DFSURWF1 DD DISP=SHR, DSN=dfsurwf1.data.set  //DFSURWF2 DD UNIT=SYSDA, SPACE=(CYL,(5,3)),  //  DISP=(,CATLG,DELETE), DSN=dfsurwf2.data.set
//DFSURWF3 DD UNIT=SYSDA, SPACE=(CYL,(5,3)),  //  DISP=(,PASS), DSN=dfsurwf3.data.set
//DFSURCDS DD DISP=SHR, DSN=dfsurcds.data.set  //IL0101 DD DISP=SHR, DSN=tst.il0101 database DSGROUP 1
//IL0102 DD DISP=SHR, DSN=tst.il0102 database DSGROUP 2
//PLUSIN DD *
GLBL UPDATE(Y)
```

Commands and keywords

You can specify the following commands in a Prefix Resolution/Update function job. The command must be the first nonblank characters. Table 43 describes the valid commands:

Table 43 Prefix Resolution/Update function commands (part 1 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLBL</td>
<td>Can be used to override the global options module. Not all options can be overridden. If present, the GLBL command must be the first statement specified.</td>
</tr>
</tbody>
</table>
Table 44 presents an overview of the options that you can set for the Prefix Resolution/Update function.

Table 44  Prefix Resolution/Update keywords  (part 1 of 2)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABNDNO</td>
<td>(Y) (N)</td>
<td>(YES)</td>
<td>406</td>
<td>Cause an abend if no output records are written to any output file. Specify N to permit condition to occur.</td>
</tr>
<tr>
<td>ALLOWLC</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>407</td>
<td>Allow logical children with no logical parents. Abend if condition occurs with default.</td>
</tr>
<tr>
<td>ALLOWLP</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>407</td>
<td>Allow logical parents with no logical children. Do not abend if condition occurs with default.</td>
</tr>
<tr>
<td>DBRC</td>
<td>(Y) (N)</td>
<td>none</td>
<td>423</td>
<td>Invoke DBRC when there is no region controller. Specify N to invoke only by region controller.</td>
</tr>
<tr>
<td>DESC</td>
<td>(desccode)</td>
<td>(7)</td>
<td>490</td>
<td>Consult MVS Supervisor Macros Manual under WTO and WTOR macros for proper designation.</td>
</tr>
<tr>
<td>FIXED</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>437</td>
<td>Allow fixed-length sorting of the logical relationship records.</td>
</tr>
<tr>
<td>FORCelp</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>439</td>
<td>Change counter fields where you are deleting LC segment during reorganization process.</td>
</tr>
<tr>
<td>GSGNAME</td>
<td>valid GSG name</td>
<td>name used in system generation</td>
<td>442</td>
<td>Use the specified global service group (GSG) for remote site recovery (RSR).</td>
</tr>
<tr>
<td>HIPO</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>446</td>
<td>Pass output from Parent-Child Resolution directly to Logical Twin Resolution using virtual storage internal buffering.</td>
</tr>
<tr>
<td>LIM</td>
<td>(value) max 9999</td>
<td>(500)</td>
<td>478</td>
<td>Notification given if orphan logical children or parents without children are found. Parameter determines how frequently to issue the notification.</td>
</tr>
<tr>
<td>NOTIFY</td>
<td>(userID)</td>
<td>none</td>
<td>495</td>
<td>Specify TSO users to be notified when events occur.</td>
</tr>
</tbody>
</table>
Global options modules

You can specify options for the Prefix Resolution/Update function at the levels shown in Table 45 on page 281.

**NOTE**

You can override any option specified in an options module by including that option in the PLUSIN control statement data set.

### Table 44  Prefix Resolution/Update keywords (part 2 of 2)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PADCHAR</td>
<td>PADCHAR(x)</td>
<td>PADCHAR(X'00')</td>
<td>503</td>
<td>Specify character to use to pad segments that are extended between the unload and reload.</td>
</tr>
<tr>
<td></td>
<td>PADCHAR('x')</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PADCHAR(‘x’)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PADCHAR(C‘x’)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PADCHAR(X‘xx’)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PADCHAR(X’00’)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDX</td>
<td>(pdxName)</td>
<td>none</td>
<td>519</td>
<td>Use a PDX to store statistical data captured during processing.</td>
</tr>
<tr>
<td>PRTOLC</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>521</td>
<td>Print logical child orphans that are logical child records with no logical parent.</td>
</tr>
<tr>
<td>PRTOLP</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>521</td>
<td>Print logical parent orphans that are logical parent record with no logical children.</td>
</tr>
<tr>
<td>ROUTCDE</td>
<td>(routeCode)</td>
<td>(1,7,11)</td>
<td>490</td>
<td>For discussion, see MVS Supervisor Macros Manual under WTO and WTOR macros.</td>
</tr>
<tr>
<td>SORTIDX</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>544</td>
<td>Run Secondary Index Resolution function during Parent-Child Resolution.</td>
</tr>
<tr>
<td>SORTIN</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>544</td>
<td>Create IMS-format records and read the IMS-format work file on SORTIN.</td>
</tr>
<tr>
<td>UNIT</td>
<td>(unitName)</td>
<td>(TAPE)</td>
<td>559</td>
<td>Specify the unit name to use when dynamically allocating the PRPURIDX work data set.</td>
</tr>
<tr>
<td>UPDATE</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>564</td>
<td>Run concurrent prefix update during the output phase of DFSURG30/DFSURG10. This completes the prefix update and precludes creation of a DFSURWF3 data set for updated data sets.</td>
</tr>
<tr>
<td>XFIXED</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>575</td>
<td>Allow fixed-length sort of index records.</td>
</tr>
</tbody>
</table>
You can combine the global options module, the PDX options member, and command methods for specifying options for maximum efficiency. You can use the PRPPARMS options module to specify the values you want to use for most of the databases at your site. For particular keywords you want to use only on a single IMS system (for example, a test system), you can use the PRP@imsi options module. For databases that need values different from those you specified in the PRPPARMS or PRP@imsi modules, you can create a PDX options member for those DBDs. For jobs that require values different from those you usually use for the database, you can specify the values in the PLUSIN data set.

### Create using ISPF interface

You can create and modify the PRPPARMS and PRP@imsi global options modules by using the Prefix Resolution/Update function ISPF interface. You can also create the DBD-level global members that are stored in the PDX. Figure 70 on page 282 shows the hierarchy of the Prefix Resolution/Update function ISPF interface.

### Table 45  Ways to specify options for the Prefix Resolution/Update function

<table>
<thead>
<tr>
<th>Module</th>
<th>Used for</th>
<th>Resides in</th>
<th>Created by</th>
</tr>
</thead>
<tbody>
<tr>
<td>internal defaults</td>
<td>all jobs that don’t specify an options module or options in PLUSIN</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>PRPPARMS</td>
<td>all jobs at your installation</td>
<td>load library</td>
<td>ISPF panels or #DBUGLBL batch job</td>
</tr>
<tr>
<td>PRP@imsi</td>
<td>all jobs run on a particular IMS system (for example, a test system)</td>
<td>load library</td>
<td>ISPF panels or #DBUGLBL batch job</td>
</tr>
<tr>
<td>DFLT member of PDX</td>
<td>all jobs, all jobs run on a particular IMS system, or all jobs for a particular database, depending on how the options module was created</td>
<td>PDX</td>
<td>an earlier release of the product</td>
</tr>
<tr>
<td>dbname member of PDX</td>
<td>all jobs for a particular database</td>
<td>PDX member</td>
<td>ISPF panels</td>
</tr>
<tr>
<td>PLUSIN</td>
<td>the job where the options are specified</td>
<td>JCL</td>
<td>adding keywords to PLUSIN control statement data set in reload job</td>
</tr>
</tbody>
</table>

---

Create using ISPF interface
The following is a checklist for using the ISPF interface to set options.

1. Access the BMC Software IMS DBU Product Selection Menu.

2. Enter S to select your MAXM Reorg solution.

3. Enter 4 to select the Prefix Resolution/Update Function Primary Menu.
4 On the Prefix Resolution/Update Function Primary Menu, type the name of your Prefix Resolution/Update Function execution library in the Prefix Resolution/Update function Global Options Library field, and enter option 3.

5 On the Prefix Resolution/Update function Install panel, enter the name of the options module you want to create or modify.

6 On the Selection Menu, enter option 1 to prompt through the specification process.

7 On each Parameters panel, type the values for the options you want to change, and enter the END command to go to the next panel.

8 When the Selection Menu reappears, enter the END command to link-edit the options module.

Create using batch job

You can create the PRPPARMS or PRP@imsi global options modules by executing the job in member #MXEGLBL of the sample library.

Commands

The command must be the first nonblank characters. The valid commands are described in “Commands and keywords” on page 278. For information about control statement syntax, see “PLUSIN control statement syntax” on page 404.

Valid keywords

Table 46 shows the keywords and the commands to which they apply.

<table>
<thead>
<tr>
<th>Keyword</th>
<th>GLBL</th>
<th>INDX</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDBD</td>
<td>Required</td>
<td></td>
<td>source DBD name</td>
</tr>
<tr>
<td>SSEG</td>
<td>Required</td>
<td></td>
<td>source segment name</td>
</tr>
<tr>
<td>XDFLD</td>
<td>Required</td>
<td></td>
<td>XDFLD name in the source segment</td>
</tr>
<tr>
<td>PARMBLK</td>
<td>Optional</td>
<td></td>
<td>override PARMS member</td>
</tr>
<tr>
<td>LIM</td>
<td>Optional</td>
<td></td>
<td>limit between orphan messages</td>
</tr>
<tr>
<td>ROUTCDE</td>
<td>Optional</td>
<td></td>
<td>MCS routing codes</td>
</tr>
<tr>
<td>DESC</td>
<td>Optional</td>
<td></td>
<td>MCS descriptor codes</td>
</tr>
<tr>
<td>SORTIN</td>
<td>Optional</td>
<td></td>
<td>read IMS records on SORTIN</td>
</tr>
<tr>
<td>HIPO</td>
<td>Optional</td>
<td></td>
<td>High Performance Option</td>
</tr>
</tbody>
</table>
The Prefix Resolution/Update function generates the reports shown in Table 47. For more information about the reports, refer to the page number shown.

**Table 46  Batch control statement keywords and commands (part 2 of 2)**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>GLBL</th>
<th>INDX</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORTDD</td>
<td>Optional</td>
<td></td>
<td>second sort ddname prefix</td>
</tr>
<tr>
<td>BUFSIZE</td>
<td>Optional</td>
<td></td>
<td>chain buffer size</td>
</tr>
<tr>
<td>NBUF</td>
<td>Optional</td>
<td></td>
<td>number of chain buffers</td>
</tr>
<tr>
<td>SIZE</td>
<td>Optional</td>
<td></td>
<td>first sort main storage size</td>
</tr>
<tr>
<td>PDX</td>
<td>Optional</td>
<td></td>
<td>PDX name</td>
</tr>
<tr>
<td>INST</td>
<td>Optional</td>
<td></td>
<td>installation name</td>
</tr>
<tr>
<td>ABNDNO</td>
<td>Optional</td>
<td></td>
<td>abend if no output</td>
</tr>
<tr>
<td>FIXED</td>
<td>Optional</td>
<td></td>
<td>use fixed length sorting (LREL)</td>
</tr>
<tr>
<td>XFIXED</td>
<td>Optional</td>
<td></td>
<td>use fixed length sorting (INDX)</td>
</tr>
<tr>
<td>SORTIDX</td>
<td>Optional</td>
<td></td>
<td>sort secondary index</td>
</tr>
<tr>
<td>UNIT</td>
<td>Optional</td>
<td></td>
<td>unit name for dynamic allocation</td>
</tr>
<tr>
<td>PRTOLP</td>
<td>Optional</td>
<td></td>
<td>print orphan logical parents</td>
</tr>
<tr>
<td>PRTOLC</td>
<td>Optional</td>
<td></td>
<td>print orphan logical children</td>
</tr>
<tr>
<td>ALLOWLP</td>
<td>Optional</td>
<td></td>
<td>allow orphan logical parents</td>
</tr>
<tr>
<td>ALLOWLC</td>
<td>Optional</td>
<td></td>
<td>allow orphan logical children</td>
</tr>
<tr>
<td>TRACE</td>
<td>Optional</td>
<td></td>
<td>print record trace during process</td>
</tr>
<tr>
<td>NOTIFY</td>
<td>Optional</td>
<td></td>
<td>user IDs to be notified</td>
</tr>
<tr>
<td>UPDATE</td>
<td>Optional</td>
<td></td>
<td>use Concurrent Prefix Update</td>
</tr>
<tr>
<td>DBRC</td>
<td>Optional</td>
<td></td>
<td>Force DBRC</td>
</tr>
<tr>
<td>FORCELP</td>
<td>Optional</td>
<td></td>
<td>Force update of LP counter when the field is zero</td>
</tr>
</tbody>
</table>

**Reports**

The Prefix Resolution/Update function generates the reports shown in Table 47. For more information about the reports, refer to the page number shown.

**Table 47  Prefix Resolution/Update function reports**

<table>
<thead>
<tr>
<th>Report</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Set Buffer Statistics</td>
<td>587</td>
</tr>
<tr>
<td>Listing of PLUSIN Control Statements</td>
<td>601</td>
</tr>
<tr>
<td>Logical Relationship Summary</td>
<td>601</td>
</tr>
<tr>
<td>Output Data Set Summary</td>
<td>605</td>
</tr>
<tr>
<td>Process Summary</td>
<td>607</td>
</tr>
<tr>
<td>Reports for DBIL Type Reorganization</td>
<td>608</td>
</tr>
<tr>
<td>Reports for DBR Type Reorganization</td>
<td>609</td>
</tr>
<tr>
<td>Secondary Index Relationship Summary</td>
<td>611</td>
</tr>
</tbody>
</table>
Performance considerations

You might be able to improve performance by using the features and options described in Table 48.

Table 48  Features and options for performance improvement

<table>
<thead>
<tr>
<th>Feature or option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORTIDX(N)</td>
<td>When you specify SORTIDX(N), the Prefix Resolution and Prefix Update functions do not sort the secondary index records that are written to the DFSURIDX data set. Using SORTIDX(N) option dramatically decreases the amount of sort work space needed, as well as the CPU and elapsed time, for the prefix resolution step. You can use the unsorted secondary index records as input to the Secondary Index Resolution function (PRPURG40) or the Index Build function.</td>
</tr>
<tr>
<td>BMC–format records</td>
<td>You can invoke the Work File Generator to write BMC-format work file records during load, reload, or scan processes. The Prefix Resolution and Prefix Update functions also write BMC–format records to pass between the Parent-Child Resolution and Logical Twin Resolution phases of the Two-step Prefix Resolution function. BMC–format records are shorter than IMS-format records, so less sort work space and fewer I/O operations are required to process them.</td>
</tr>
<tr>
<td>HIPO(N)</td>
<td>When you specify HIPO(N), the Prefix Resolution function generally runs two to four times faster than the IBM Prefix Resolution utility and uses approximately 40 to 50 percent less sort work space.</td>
</tr>
<tr>
<td>HIPO(Y)</td>
<td>When you specify HIPO(Y), the Prefix Resolution function runs up to 10 times faster than the IBM Prefix Resolution utility and uses about the same amount of sort work space as the IBM utility.</td>
</tr>
<tr>
<td>FIXED(Y)</td>
<td>When you specify FIXED(Y) or XFIXED(Y), sort performance is improved, but more sort work space is required.</td>
</tr>
<tr>
<td>FORCELBP(Y)</td>
<td>When you specify FORCELBP(Y) during a DBIL-type reorganization, it degrades the performance of the Prefix Update program.</td>
</tr>
<tr>
<td>No DCB information</td>
<td>The Prefix Resolution and Prefix Update functions calculate the optimum DCB information for the device on which the data set will reside. Generally, you should not provide DCB information for any output data set.</td>
</tr>
<tr>
<td>UPDATE(Y)</td>
<td>When you specify UPDATE(Y), the Prefix Resolution and Prefix Update functions resolve and update the logical relationship information in the same job step, at a decrease in processing time of about 10 percent over running the Prefix Resolution and Prefix Update function in two steps. The primary source for improvement is in the reduction of I/O for writing the DFSURWF3 (or equivalent) data set.</td>
</tr>
</tbody>
</table>
Other prefix resolution and update functions

This section describes other prefix resolution and update functions available.

IMS-compatible Prefix Resolution function

The IMS-compatible Prefix Resolution function runs two to four times faster and uses 40 percent less sort work space than the IMS Prefix Resolution utility. You can use your existing prefix resolution JCL with minor changes.

Figure 71 shows the system flow of the IMS-compatible Prefix Resolution function.
The IMS-compatible Prefix Resolution function works as follows:

1. The Prefix Resolution function reads the DBD library to get information about the logical relationships.

2. The Prefix Resolution function reads the control data set (DFSURCDS) that was generated by the IMS Prereorganization utility.

3. If DBRC is active, the Prefix Resolution function reads and writes recovery control information to the RECON data sets.

4. The DFSURWF1 or SORTIN data set supplies the logical relationship records and can supply the secondary index records that were created during the database load, reload, or scan.

   The DFSURWF1 data set always contains IMS-format logical relationship records and can also contain secondary index (type 40) records.

   The SORTIN data set can contain either (but not both) of the following record types:

   - BMC-format logical relationship records. The SORTIN data set cannot contain secondary index records when it contains BMC-format logical relationship records. Specify SORTIN(N) to indicate that the records are in BMC format.

   - IMS-format logical relationship records and secondary index records. Specify SORTIN(Y) to indicate that the records are in IMS format.

5. The IMS-compatible Prefix Resolution function (program name DFSURG10 or PRPURG10) consists of two sort phases, PRPURG20 and PRPURG30, that run serially (one at a time) rather than concurrently. To run the phases serially, specify HIPO(N). Both phases invoke your site’s sort utility and use the SORTWK*nn data sets:

   - The first phase, Parent-Child Resolution (program name PRPURG20), sorts logical child relationship records with their corresponding logical parent records and resolves the logical child and logical parent pointers. This phase creates an intermediate work file data set (DFSURWF2) containing partially resolved work file records and passes this data set to the second sort phase.

   - The second phase, Logical Twin Resolution (program name PRPURG30), sorts the records into physical database sequence (within the data set) and resolves any logical twin forward and logical twin backward pointers. This phase creates and sorts logical relationship work file (DFSURWF3) records, which you use as input to the Prefix Update function (PRPURGP0). You can run the Prefix Update function concurrently with Logical Twin Resolution if you supply a DD statement defining the database data set to be updated and specify UPDATE(Y).
6. The Prefix Resolution function extracts any index records from the DFSURWF1 or SORTIN data set during the input phase of PRPURG20:

- If you specify SORTIDX(Y), the Prefix Resolution function sorts the records by invoking the Secondary Index Resolution function (PRPURG40). It creates a temporary PRPURIDX work data set for use during the sort.

- If you specify SORTIDX(N), you must process the index records using the Secondary Index Resolution function (PRPURG40), the Index Build function, or the IMS Prefix Resolution utility.

7. The Prefix Resolution function captures and reports statistics. It writes these reports to the SYSPRINT data set, and it can write them to the PDX data set.

**JCL requirements**

Figure 72 shows the JCL requirements for the IMS-compatible Prefix Resolution function.

---

**Figure 72 IMS-compatible Prefix Resolution function JCL**

```plaintext
//JOBNAME JOB ',USER',CLASS=A,MSGCLASS=X,NOTIFY=USER
//***********************************************************
//PREFIX EXEC PGM=DFSURG10,REGION=0M
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// SYSPRINT DD SYSOUT=* 
//SYSOUT DD SYSOUT=* 
//PLUSOUT DD SYSOUT=* 
//PLUSLIST DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 
//IMS DD DSN=IMSVS.DBDLIB,DISP=SHR 
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(5,3))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(5,3))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(5,3))
//DFSURWF1 DD DISP=SHR,DSN=DFSURWF1.DATA.SET
//DFSURWF2 DD UNIT=SYSDA,SPACE=(CYL,(5,3)).
// DFSURWF3 DD UNIT=SYSDA,SPACE=(CYL,(5,3)).
// DFSURCDS DD DISP=SHR,DSN=DFSURCDS.DATA.SET
//PLUSIN DD *
GLBL SORTIN(Y) HIPO(N) UPDATE(N)
```
Two-step Prefix Resolution function

For more information about the DD statements, see Chapter 13, “JCL requirements.” The following DD statements are specific to the IMS-compatible Prefix Resolution function:

**EXEC**

Required. Must specify DFSURG10 or PRPURG10. Do not execute the IMS-compatible Prefix Resolution function under the IMS region controller (PGM=DFSRRC00). BMC recommends a region size of 0M. The Prefix Resolution function does not accept the CORE parameter on this statement.

**physical database DD**

Optional. Defines the work data set that the Prefix Resolution function creates during the Logical Twin Resolution phase. This data set contains DFSURWF3-format logical relationship records for a physical database as identified by the physical database name. When you include this DD statement, the Prefix Resolution function writes all logical child and logical parent records for the physical database to this data set, unless you specify physical data set DD statements to split the records.

**physical data set DD**

Required if you specify UPDATE(Y); otherwise, optional.

- If you specify UPDATE(Y) option, this statement defines a database data set that the Prefix Resolution function updates using concurrent prefix update.

- If you specify UPDATE(N), this statement defines the work data set created during the Logical Twin Resolution phase. (This data set contains DFSURWF3-format logical relationship records for a physical data set as identified by the physical data set ddname.) When you include this DD statement, the Prefix Resolution function writes all logical child and logical parent records for the physical data set to this data set.

**Two-step Prefix Resolution function**

The Two-step Prefix Resolution function is similar to the IMS-compatible Prefix Resolution function. The main difference is that the Parent-Child Resolution phase (program name PRPURG20) and Logical Twin Resolution phase (program name PRPURG30) are executed in separate job steps. The JCL provides the transition from the first sort to the second sort, instead of the program providing this transition.

If a failure occurs during the Logical Twin Resolution (PRPURG30) phase, you can restart the job at the beginning of the PRPURG30 step instead of having to reprocess the Parent-Child Resolution (PRPURG20) phase.
Figure 73 shows the system flow of the Two-step Prefix Resolution function.

**Figure 73  Two-step Prefix Resolution function system flow**
The Two-step Prefix Resolution function works as follows:

1. The Two-step Prefix Resolution function reads the DBD library to get information about the relevant logical relationships.

2. The Two-step Prefix Resolution function reads the control data set (DFSURCDS) that was generated by the IMS Prereorganization utility.

3. If DBRC is active, the Two-step Prefix Resolution function reads and writes recovery control information to the RECON data sets.

4. The DFSURWF1 or SORTIN data set is input to the Parent-Child Resolution phase (PRPURG20). This data set supplies the logical relationship records and can supply the secondary index records that were created during the database load, reload, or scan. This data set can contain data that has been split on a logical child segment name or a logical parent DBD name.

   The DFSURWF1 data set always contains IMS-format logical relationship records and can also contain secondary index (type 40) records.

   The SORTIN data set can contain either (but not both) of the following record types:

   - BMC-format logical relationship records. The SORTIN data set cannot contain secondary index records when it contains BMC-format logical relationship records. Specify SORTIN(N) to indicate that the records are in BMC format.
   - IMS-format logical relationship records and secondary index records. Specify SORTIN(Y) to indicate that the records are in IMS format.

5. The Parent-Child Resolution phase (program name PRPURG20) sorts logical child relationship records with their corresponding logical parent records and resolves the logical child and logical parent pointers. It invokes your site’s sort utility and use the SORTWKnn data sets. The Parent-Child Resolution phase creates partially resolved work file records.

6. The Parent-Child Resolution phase extracts any index records from the DFSURWF1 or SORTIN data set.

   - If you specify SORTIDX(Y), the Two-step Prefix Resolution function sorts the records by invoking the Secondary Index Resolution function (PRPURG40). It creates a temporary PRPURIDX work data set for use during the sort.
   - If you specify SORTIDX(N), you must process the index records using the Secondary Index Resolution function (PRPURG40), the Index Build function, or the IMS Prefix Resolution utility.
7. The output of the Parent-Child Resolution step can be split on a physical DBD name or on a data set ddname. Any records not selected using one of the above fields are written to the DFSURWF2 data set.

8. The Logical Twin Resolution step (program name PRPURG30) sorts the records into physical database sequence (within the data set) and resolves any logical twin forward and backward pointers. This step invokes your site’s sort utility and uses the SORTWKnn data sets.

9. The output of the Logical Twin Resolution step can be split on the ddname for the data set group in which the physical segment reside or on the physical DBD name group in which the segment resides. Any records not selected using one of the above fields are written to the DFSURWF3 data set. Each data set created as output from PRPURG30 can be processed individually by the Prefix Update function (PRPURGP0).

You can run the Prefix Update function concurrently with PRPURG30 if you supply a DD statement defining the data set to be updated and specify UPDATE(Y). To prevent unnecessary DASD arm contention when running the updates concurrently, ensure that jobs for data sets in the same volume are not processed concurrently.

10. The Two-step Prefix Resolution function captures and reports statistics. It writes these reports to the SYSPRINT data set, and it can write them to the PDX data set.

**JCL requirements**

Figure 74 shows the JCL requirements for the Two-step Prefix Resolution function. Both steps (PRPURG20 and PRPURG30) must be processed, even if there are no logical twin pointers, because PRPURG30 also sorts the data into sequence by RBA within data set within database. This improves performance for the Prefix Update utility (DFSURGP0).

**Figure 74  Two-step Prefix Resolution function sample JCL (part 1 of 2)**

```plaintext
//JOBNAME  JOB ',USER',CLASS=A,MSGCLASS=X,NOTIFY=USER
//***********************************************************
//PREFIX1 EXEC PGM=PRPURG20,REGION=0M
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//SYSPRINT DD SYSOUT=*  //SYSOUT DD SYSOUT=*
//PLUSDUMP DD SYSOUT=* (dump orphan LC records)
//SYSUDUMP DD SYSOUT=*  //IMS DD DSN=IMSVS.DBDLIB,DISP=SHR
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(5,3))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(5,3))
//SORTWK03 DD UNIT=SYSDA,SPACE=(CYL,(5,3))
//SORTIN DD DISP=SHR,DSN=prpurwf1.data.set
//DFSURWF2 DD UNIT=SYSDA,SPACE=(CYL,(5,3)).
```
Two-step Prefix Resolution function

Chapter 10 Working with logical relationships

For more information about the DD statements, see Chapter 13, “JCL requirements.”

The following DD statements are specific to the Two-step Prefix Resolution function:

**Parent-Child Resolution step JCL statements**

The following JCL statements are used to run the Parent-Child Resolution function:

```plaintext
EXEC

Required. Must specify PRPURG20. BMC recommends a region size of 0M. The Two-step Prefix Resolution function does not accept the CORE parameter on this statement.

**DFSURWF1 DD**

Required if you do not supply a SORTIN DD statement. If the data has been split, you must ensure that supply all sources of a logical relationship to the same PRPURG20 run (logical parent and logical child); failure to do so might result in erroneous output, orphan logical child segments, and/or logical parent segments with no logical child segments. For more information, see “DFSURWF1 DD” on page 364.

**SORTIN DD**

Required if you do not supply a DFSURWF1 DD statement. records. If the data has been split, you must ensure that you supply all sources of a logical relationship to the same PRPURG20 run (logical parent and logical child); failure to do so might result in erroneous output, orphan logical child segments, and/or logical parent segments with no logical child segments. See “SORTIN DD” on page 377.

Figure 74 Two-step Prefix Resolution function sample JCL (part 2 of 2)

```
physical database DD

Optional. Defines the work data set the Two-step Prefix Resolution function creates during the Parent-Child Resolution phase. It contains BMC-format logical relationship records for a given physical database as identified by the physical database name. When you include this DD statement, the Two-step Prefix Resolution function writes all logical child and logical parent records for the physical database to this data set, unless you specify physical data set DD statements to further split the records.

physical data set DD

Optional. Defines the work data set created during the Parent-Child Resolution step that will contain BMC-format logical relationship records for a given physical data set as identified by the physical data set ddname (DSGROUP). When you include this DD statement, the Two-step Prefix Resolution function writes all logical child and logical parent records for the physical database to this data set.

Logical Twin Resolution step JCL statements

The Logical Twin Resolution step (PRPURG30) takes DFSURWF2 output records from PRPURG20 or from DFSURG10 steps that abended in the Logical Twin Resolution phase and sorts them by physical parent database, physical data set, and RBA within the data set. The Logical Twin Resolution step also resolves logical twin pointers. This step produces one or more data sets that you can use as input to the Prefix Update function or the IMS Prefix Update utility.

The following JCL statements are used to run the Logical Twin Resolution step:

EXEC

Required. Must specify PRPURG30. BMC recommends a region size of 0M. the Two-step Prefix Resolution function does not accept the CORE parameter on this statement.

SORTIN DD

Required. Defines the work data set created during the Parent-Child Resolution phase (PRPURG20) containing BMC-format DFSURWF2 logical relationship records. The logical relationship records for all logical parent and logical child segments that exist in a physical database must be concatenated to form a single input stream for Logical Twin Resolution. Logical parents are written to their physical database, and logical children are written to their physical database.

You must supply BMC-format DFSURWF2 records. IMS-format DFSURWF2 records are not valid for the Logical Twin Resolution step.
physical database DD

Optional. Defines the work data set that the Two-step Prefix Resolution function creates during the Logical Twin Resolution phase. This data set contains DFSURWF3-format logical relationship records for a physical database as identified by the physical database name. When you include this DD statement, the Two-step Prefix Resolution function writes all logical child and logical parent records for the physical database to this data set, unless you specify physical data set DD statements to split the records.

physical data set DD

Required if you specify UPDATE(Y); otherwise, optional.

- If you specify UPDATE(Y), this statement defines a database data set that the Two-step Prefix Resolution function updates using concurrent prefix update.

- If you specify UPDATE(N), this statement defines the work data set created during the Logical Twin Resolution phase. (This data set contains DFSURWF3-format logical relationship records for a physical data set as identified by the physical data set ddname or data set group name.) When you include this DD statement, the Two-step Prefix Resolution function writes all logical child and logical parent records for the physical data set to this data set.

Stand-alone Prefix Update function

The Prefix Update function (program name PRPURGP0) replaces the IMS utilities that update the pointers in the prefixes of logically related segments to establish the logical relationships. You can run the Prefix Update function as a stand-alone utility, under the IMS region controller, or concurrently as the output of any Prefix Resolution function.

When run as a stand-alone utility, the Prefix Update function (PRPURGP0) uses as input the sorted logical relationship records that were written by the Logical Twin Resolution phase (PRPURG30). The Prefix Update function places the correct logical relationship pointers in the prefixes of the logically related databases.

Figure 75 on page 296 shows the system flow of the Prefix Update function when run as a stand-alone utility.
The Prefix Update function works as follows when run as a stand-alone utility:

1. The Prefix Update function reads the DBD library to get information about the relevant logical relationships.

2. The Prefix Update function reads the control data set (DFSURCDS) that was generated by the IMS Prereorganization utility.

3. If DBRC is active, the Prefix Update function reads and writes recovery control information to the RECON data sets.

4. Input is the sorted logical relationship work file (DFSURWF3) records from the Logical Twin Resolution function (PRPURG30).

5. Output is the updated database data sets.

6. The Prefix Update function captures and reports statistics. It writes these reports to the SYSPRINT data set, and it can write them to the PDX data set.

**JCL requirements**

Figure 76 on page 297 shows the JCL requirements for running the Prefix Update function as a stand-alone utility. The input is a DFSURWF3-format data set from the Logical Twin Resolution phase of a Prefix Resolution function, and the output is the physical database data sets.
PRPURGP0 requires the DFSURCDS data set. PRPURGP0 does no logging, but it does have an interface to DBRC. If DBRC is active and the database has been registered, the Image Copy Needed (ICNEEDED) flag is turned on for each database data set that has been updated. Take an image copy of the database after the prefix update phase completes successfully.

**Figure 76  Stand-alone Prefix Update function JCL**

```
//JOBNAME JOB ,USER',CLASS=A,MSGCLASS=X,NOTIFY=USER
//*******************************************************************************
//UPDATE EXEC PGM=PRPURGP0,REGION=0M
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//SYSPRINT DD SYSPUT=*
//PLUSOUT DD SYSPUT=*
//PLUSLIST DD SYSPUT=*
//SYSUDUMP DD SYSPUT=*,DCB=BLKSIZE=120
//IL0101 DD DISP=SHR,DSN=tst.il0101 database DSGROUP 1
//IL0102 DD DISP=SHR,DSN=tst.il0102 database DSGROUP 2
//DFSURWF3 DD DISP=SHR,DSN=tst dfsurwf3 output of prefix resolution
//IL01 DD DISP=SHR,DSN=tst_il01 partition on DBD IL01
//IL01F DD DISP=SHR,DSN=tst_il01f partition on SEGM IL01F
//DFSURCDS DD DISP=SHR,DSN=tst dfsurcds
```

For more information about the DD statements, see Chapter 13, “JCL requirements.”

The following DD statements are specific to the Stand-alone Prefix Update function:

**EXEC**

Required. Must specify DFSURGP0 or PRPURGP0. You can execute the Prefix Update function under the IMS region controller (PGM=DFSRRRC00). BMC recommends a region size of 0M.

**ddname DD**

Required. Defines the database data sets to be updated by the Prefix Update function.

---

**Secondary Index Resolution function**

To handle secondary index records in the most efficient manner, use the Index Build function. It performs all the processing necessary to rebuild secondary indexes after a database reorganization. The Index Build function does not require you to unload and reload the database to create secondary indexes.

Because the Index Build function creates secondary indexes, the IMS HISAM Unload/Reload utilities are not needed.
The Index Build function can process secondary index records from the following sources:

- DFSURWF1 file written by the load or reload
- DFSURIDX file split from the DFSURWF1 file by the Work File Splitter
- sorted DFSURIDX data set. The Prefix Resolution/Update function sorts the records when you specify SORTIDX(Y). Use this data set as input to the Index Build function with the DFSURIDX DD statement to indicate that the records do not need to be sorted.
- unsorted DFSURIDX data set. The Prefix Resolution/Update function does not sort the records when you specify SORTIDX(N). Use this data set as input to the Index Build function with the DFSURWF1 DD statement to indicate that the records need to be sorted.

The Secondary Index Resolution function (program name PRPURG40) provides compatibility with the IMS Prefix Resolution utility; it is not intended to replace the Index Build function. You can use the Index Build function rather than the Prefix Resolution/Update function to process the secondary index work records in the DFSURWF1 data set.

The Secondary Index Resolution function (PRPURG40) processes only IMS-format (DFSURWF1) secondary index (type 40) records. Output is an IMS-format (DFSURIDX) data set containing sorted records. These records must be processed by the HISAM Unload and Reload utilities or the Index Build function to create a secondary index.

Figure 77 shows the system flow of the Secondary Index Resolution function.

**Figure 77  Secondary Index Resolution function system flow**
The Secondary Index Resolution function (PRPURG40) works as follows:

1. The only acceptable input is IMS-format work file records. Specify the input in the DFSURWF1 data set; however, if you have specified SORTIN(Y), specify the input work file records in the SORTIN data set.

You can use the Work File Splitter to split the secondary index work records before you pass them to PRPURG40.

2. PRPURG40 requires a control data set (DFSURCDS) that references all source databases for all secondary indexes to be processed. The indexes themselves do not need to be referenced.

3. PRPURG40 invokes your site’s sort program.

### JCL requirements

Figure 78 shows the JCL requirements for the Secondary Index Resolution function.

**Figure 78  Secondary Index Resolution function JCL requirements**

```
//JOBNAME JOB , 'USER', CLASS=A, MSGCLASS=X, NOTIFY=USER
//***********************************************************
//PREFIX EXEC PGM=PRPURG40, REGION=0M
//STEPLIB DD DISP=SHR, DSN=bmc.xxx.load
// DD DISP=SHR, DSN=IMSVS.RESLIB
//SYSPRINT DD SYSOUT=* 
//SYSOUT DD SYSOUT=*
//PLUSOUT DD SYSOUT=* 
//PLUSLIST DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//IMS DD DSN=DISP=SHR, IMSVS.DBDLIB
//SORTWK01 DD UNIT=SYSDA, SPACE=(CYL,(5,3))
//SORTWK02 DD UNIT=SYSDA, SPACE=(CYL,(5,3))
//SORTWK03 DD UNIT=SYSDA, SPACE=(CYL,(5,3))
//DFSURWF1 DD DISP=SHR, DSN=dfsurwf1.data.set
//DFSURIDX DD UNIT=SYSDA, SPACE=(CYL,(5,3)),
// DISP=(, CATLG,DELETE), DSN=dfsuridx.data.set
//DFSURCDS DD DISP=SHR, DSN=dfsurcds.data.set
```

For more information about the DD statements, see Chapter 13, “JCL requirements.”

The following DD statements are specific to the Secondary Index Resolution function:

**EXEC**

Required. Must specify PRPURG40 BMC recommends a region size of 0M. The Secondary Index Resolution function does not accept the CORE parameter on this statement.
**DFSURIDX DD**

Required. Defines the work data set that contains all sorted secondary index records not specifically selected out using the secondary index physical DBD name as described below. The output of this phase is the input to the HISAM Unload utility, which extracts the secondary index for loading. If you use this data set as input to the Index Build function, describe it to the Index Build function with the DFSURIDX DD statement rather than the DFSURWF1 DD statement.

**secondary index database DD**

Optional. Defines the work data set created during the Secondary Index Resolution function that will contain the sorted secondary index work records for a given secondary index as indicated by the secondary index DBD name on the DD statement. The output of this phase is the input to the HISAM Unload utility, which will extract the secondary index for loading. If you input this data set to the Index Build function, describe it to the Index Build function with the DFSURIDX DD statement rather than the DFSURWF1 DD statement.

---

**Work File 1 Generator**

The BMC Work File 1 (WF1) Generator (PRPDSEH0) replaces the IBM-supplied WF1 Generator (DFSDSEH0).

The PRPDSEH0 module has interfaces to the IMS and BMC utilities that create logical relationship work file records. The WF1 Generator can create BMC-format work file records or IMS-format work file records.

---

**WARNING**

BMC-format work file records are not compatible with IMS-format work file records. You can only use them as input to Prefix Resolution/Update function.

---

The WF1 Generator can split the records if you provide the appropriate DD statements. For more information, see “Work File Splitter” on page 304.

The following modules are discussed in this chapter:

- DFSDSEH0—IMS WF1 Generator
- PRPDSEH0—BMC WF1 Generator; DFSDSEH0 is an alias
- DBUDSEH1—BMC WF1 Generator subroutine
Figure 79 shows the input, output, and system flow of the WF1 Generator.

Any BMC or IMS utility that builds logical relationship work file records can use the WF1 Generator. The WF1 Generator interface works with the following utilities:

- **Reload function**

  If the PRPURG10 module is in the STEPLIB/JOBLIB concatenation or in the LINKLIST, the Reload function invokes the DBUDSEH1 module. If PRPURG10 is not available, the Reload function invokes the DFSDSEH0 module, which might be the actual IMS module or the alias of PRPDSEH0, depending on the STEPLIB/JOBLIB concatenation.

- **IMS HD Scan utility**

  The IMS Scan utility always invokes the IBM WF1 Generator (DFSDSEH0) unless the BMC WF1 Generator has been link-edited into the IMS DFSURGS0 module.

- **IMS HD Reload utility**

  The IMS Reload utility always invokes the DFSDSEH0 module. The actual module is the IBM WF1 Generator or the BMC WF1 generator, depending on the STEPLIB/JOBLIB concatenation for the reload step.
The BMC WF1 Generator can build BMC-format work file records or IMS-format work file records. If you provide a PRPURWF1 (or equivalent) DD statement, the WF1 Generator creates BMC-format work file records. If you provide only a DFSURWF1 DD statement, the WF1 Generator creates IMS-format work file records.

**WARNING**

You must always provide a DFSURWF1 DD statement. The WF1 generator always writes secondary index work file records to the DFSURWF1 data set, regardless of whether it creates BMC-format records. If you specify DFSURWF1 DD DUMMY and the database contains logical relationships, the WF1 Generator will not write any work file records or statistics.

---

**JCL requirements**

Figure 80 shows the JCL requirements for the WF1 Generator.

**Figure 80  WF1 Generator JCL requirements**

```plaintext
//RELOAD EXEC PGM=DFSRRC00,PARM='ULU,DFSURGLO,dbname'.
// REGION=OM
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load <<< required
// DD DISP=SHR,DSN=IMVS.RESLIB
//DFSRESLB DD DISP=SHR,DSN=IMVS.RESLIB
//DFSUINPT DD DISP=OLD,DSN=TST.HDUNLOAD
//DFSVSAMP DD DISP=SHR,DSN=IMVS.PROCLIB(VSAMP01)
//D1 DD DISP=OLD,DSN=TSTV.data.set.name1
//SYSPRINT DD SYSOUT=A
//DFSURCDS DD DISP=OLD,DSN=TST.CDS
//DFSURWF1 DD DISP=OLD,DSN=TST.DFSURWF1
//PRPURWF1 DD DISP=OLD,DSN=TST.PRPURWF1
```

For more information about the DD statements, see Chapter 13, “JCL requirements.”

The following DD statements are specific to the WF1 Generator function:

**Output data sets**

When the WF1 Generator creates the following output data sets, it does not honor the DCB information on the DD statements.
DFSURWF1 DD

Optional. Defines the work data set that is created during database load, reload, or scan processes. The DFSURWF1 data set contains IMS-format logical relationship and secondary index records. The WF1 Generator uses these records to resolve logical and/or secondary index relationships. All secondary index records are written to this data set.

You must provide the DFSURWF1 statement if the database contains any logical or secondary index relationships, even if you create BMC-format logical relationship records by providing a PRPURWF1 DD statement, a logical child segment DD statement, or a logical parent database DD statement.

**WARNING**

If you specify DFSURWF1 DD DUMMY, the DFSDSEH0 module is not called; no PRPURWF1 or DFSURWF1 records are written, and there are no Prefix Resolution/Update function statistics on hardcopy or in the PDX.

PRPURWF1 DD

Optional. Defines the work data set the WF1 generator creates during database load, reload, or scan. It contains BMC-format logical relationship records. Use these records as input to a Prefix Resolution function.

If you provide a PRPURWF1 DD statement, the WF1 Generator creates BMC-format records for all logical relationships. If you do not provide this statement, the WF1 Generator creates IMS-format logical relationship records for all logical relationships that are not specifically split out with a logical child segment DD statement or a logical parent database DD statement.

logical child segment DD

Optional. Defines the work data set the WF1 Generator creates during the database load, reload, or scan. It contains BMC-format logical relationship records for a logical relationship as identified by the logical child segment name. If you provide this DD statement, the WF1 Generator writes all logical child and logical parent records for the logical relationship to this data set. If the logical parent and logical child segments exist in different databases, the WF1 Generator creates a data set for each database; you must concatenate these data sets when you use them as input to the Prefix Resolution function. If you do not provide this statement, the WF1 Generator writes the records to the logical parent database DD statement (if present).
**Work File Splitter**

You can use the Work File Splitter function (program name PRPSPLIT) to separate IMS-format secondary index work file records from the logical relationship records when both types of records have been written to the DFSURWF1 data set. In addition, you can split the output of the Work File Splitter.

The Work File Splitter function splits secondary index and logical relationship records in the IMS-format work file (DFSURWF1). Figure 81 shows the system flow of the Work File Splitter function.

![Figure 81 Work File Splitter function system flow](image)

**logical parent database DD**

Optional. Defines the work data set the WF1 Generator creates during the database load, reload, or scan of the database. It contains BMC-format logical relationship records for a logical parent database as identified by the logical parent database name. If you provide this DD statement, the WF1 Generator writes all logical parent records for all segments in the logical parent database and any related logical child records (regardless of their physical database) to this data set, unless you have split specific logical relationships by providing the logical child segment name. If you do not provide this statement, the WF1 Generator writes the records to the PRPURWF1 DD statement (if present).
The Work File Splitter function works as follows:

- Logical relationship work records are split by logical parent database name. If you provide a DD statement that uses the logical parent DBD name as the ddname, all logical relationship records that have logical parents in that database are written to the specified data set. All logical relationship records not specifically split are written to the DFSURWF2 data set.

- Secondary index records are split by secondary index DBD name. If you provide a DD statement that uses the secondary index DBD name as the ddname, all secondary index records for that index are written to the specified data set. All secondary index records not specifically split are written to the DFSURIDX data set, if the DFSURIDX data set is provided. If no DFSURIDX data set is provided, secondary index records that are not specifically split are bypassed.

---

**NOTE**

If you specify PART(N) or accept it as the default, the Work File Splitter function does not split logical relationship records.

---

### JCL requirements

Figure 82 shows the JCL requirements for the Work File Splitter function.

---

**Figure 82  Work File Splitter function JCL requirements**

```plaintext
//JOBNAME JOB , 'USER',CLASS=A,MSGCLASS=X,NOTIFY=USER
//***************************************************************
//SPLIT EXEC PGM=PRPSPLIT,REGION=0M
//STEPLIB DD DISP=SHR,DSN=bmc.xxx.load
//   DD DISP=SHR,DSN=IMSVS.RESLIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//SYSPRINT DD SYSOUT=*  
//DFSURWF1 DD DISP=SHR,DSN=dfsurwf1.data.set
//DFSURCDS DD DISP=SHR,DSN=dfsurcds.data.set
//DFSURWF2 DD DSN=unsplit.data.file,DISP=(NEW,CATLG,DELETE),
//   UNIT=TAPE, etc.
//DFSURIDX DD DSN=unsplit.index.file,etc.
//*
//* LOGICAL RELATIONSHIP RECORDS FOR DBD1 AND DBD2 ARE
//* WRITTEN TO THE FOLLOWING DATA SETS:
//* DBD1   DD DSN=split.data.dbd1,DCB=BLKSIZE=max, etc.
//* DBD2   DD DSN=split.data.dbd2,etc.
//*
//* SECONDARY INDEX RECORDS FOR SINDEX1 AND SINDEX2 ARE
//* WRITTEN TO THE FOLLOWING DATA SETS:
//* SINDEX1 DD DSN=split.data.sindex1, etc.
//* SINDEX2 DD DSN=split.data.sindex2, etc.
```
For more information about the DD statements, see Chapter 13, “JCL requirements.”
The following DD statements are specific to the Work File Splitter function:

**EXEC**

Required. Must specify PRPSPLIT. BMC recommends a region size of 0M.

**DFSURWF1 DD**

Required. BMC-format logical relationship records cannot be processed by PRPSPLIT. Multiple DFSURWF1-format data sets can be concatenated as input to PRPSPLIT.

**DFSURWF2 DD**

Defines the work data set that contains all IMS-format logical relationship records.

**logical parent database DD**

Optional. Defines the work data set that contains all IMS-format logical child and logical parent relationship records in the input data set for the logical parent database as identified by the logical parent DBD name on the DD statement. If you do not provide this statement, the Work File Splitter writes all output for a logical parent database to DFSURWF2.

**secondary index database DD**

Optional. Defines the work data set that contains the secondary index work records for a secondary index, as indicated by the secondary index DBD name on the DD statement. If you do not provide this statement, data for the secondary index is written to DFSURIDX. Use the output of this phase as input to the HISAM Unload Utility to extract the secondary index for loading. If you use the output of this phase as input to the Index Build function, use the DFSURWF1 DD statement.
This chapter describes how to use the Image Copy function to create an image copy of
the database as part of the reorganization process. This chapter contains the following
topics:

Overview ................................................................. 308
  Image Copy function with the Online Reorg function .................. 308
  Considerations and limitations ........................................ 309
JCL requirements ...................................................... 310
ICPSYSIN statements ............................................... 311
  ICPSYSIN commands ............................................... 312
  ICPSYSIN keywords ............................................... 313
User-controlled options ............................................. 317
  Global options modules .......................................... 318
  Database-specific options members ................................ 319
  Control statement keywords ..................................... 319
  Image Copy function global options modules ....................... 319
  Database-specific options modules ............................... 324
Reports ................................................................. 325
  Image copy data sets ............................................... 325
  Explicit allocation of image copy data sets ....................... 325
  Dynamic allocation of image copy data sets .................... 326
Virtual image copies ................................................ 330
Output image copy write options .................................. 331
  Access method control .......................................... 331
  Buffer pool size specification ................................. 332
  Image copy block size settings .................................. 333
POINTER CHECKER PLUS interface ................................ 334
Monitoring and reporting options .................................. 334
  Monitor the progress of the job .................................. 335
  Free space analysis information ................................ 335
  PDX history limit settings ....................................... 335
GROUP control statement .......................................... 335
  User-defined GROUP processing .................................. 336
  DBRC-defined GROUP processing ............................... 337
Overview

The Image Copy function enables you to take an image copy of the database during the reorganization.

Image Copy function with the Online Reorg function

Taking an image copy is required for the Online Reorg function. You do not need to specify the ICP keyword in an Online Reorg function job. If you specify ICP(NO), the Online Reorg function ignores the specification, forces the value to ICP(YES), and issues message BMC90454.

You may supply an ICPSYSIN DD statement with an IC or AIC command for each recoverable database on the ICPSYSIN DD statement, or the Online Reorg function will automatically create an image copy for all databases being reorganized that are registered as RECOV in DBRC.

Using the Image Copy function with the Reorg function requires appropriate settings for the following parameters:

- Specify ICP(YES) when you use the Reorg function, except when the database has logical relationships. Performing the image copy concurrently with the reorganization eliminates the outage required to take an image copy.

- Specify ICP(NO) if the database you are reorganizing has logical relationships. ICP(YES) is rejected for databases with direct logical pointers. The image copy would not be useful if it was taken before the prefix resolution and update processes were complete. If you are reorganizing databases that have logical relationships, take an image copy in a separate step after the reorganization and prefix resolution and update steps.

**NOTE**

If you specify SWAP(x,IC), the primary database and primary index are always treated as RECOV.

You can specify VIC(Y) in the ICPSYSIN control statement data set. The Reorg function supports virtual image copies.
Considerations and limitations

The following considerations and limitations apply to the use of the Image Copy function:

**Database recovery**

If you need to recover a database that was reorganized by the Online Reorg function, note that the image copy time stamp in the header of the image copy data set created by the Online Reorg function is not the same as what is posted in DBRC. The time stamp in the image copy header is created at image copy initialization, and the NOTIFY.IC is posted at the end of the reorganization for data integrity purposes. You should be aware of this situation because it can affect recovery.

RECOVERY PLUS for IMS handles this situation with no manual intervention. However, if you are using the IMS Recovery utility, the utility will issue a message indicating that the time stamps do not match. To avoid zapping the IMS Recovery utility to bypass this check, BMC recommends that you concatenate the BMC load library in your IMS Recovery utility STEPLIB concatenation (ahead of RESLIB) when you recover a database using the image copy created during the Online Reorg function reorganization. Including the BMC load library in your IMS Recovery utility STEPLIB concatenation ahead of RESLIB causes the Online Reorg function to dynamically bypass checking the time stamps (no usermod or zap is required).

**Stacked image copies**

If a database consists of multiple data set groups, has indexes, or both, the image copies cannot be stacked.

**Indexes**

If any indexes are set to NONRECOV in DBRC, they will not be image copied and a NOTIFY.UIC will be issued. DBRC must be active for index databases to be image copied.

If a secondary index is set to RECOV in DBRC, the Image Copy function attempts to take an image copy. If a secondary index is recoverable and an image copy data set cannot be allocated, either dynamically or through JCL statements, the Image Copy function fails when it attempts to take an image copy of the index.
**JCL requirements**

The Image Copy function, which is included with MAXM Reorg solutions, runs under the Online Reorg function or Reorg function only; it does not run as a stand-alone utility. The BMC Image Copy utility, which is included with the Backup and Recovery Solution for IMS product and the IMAGE COPY PLUS product, can be executed as a stand-alone utility. The Image Copy function is required for the Online Reorg function. It is optional for the Reorg function.

The Image Copy function has many of the same JCL requirements as the IMAGE COPY PLUS product from BMC. If you have existing IMAGE COPY PLUS JCL, you might be able to copy statements (such as ICPSYSIN) into the JCL for the executing the Online Reorg function or Reorg function.

The following DD statements are specific to the Image Copy function. For more information about other DD statements, see Chapter 13, “JCL requirements.”

### ICPSYSIN DD

Optional. Contains the commands and keywords for the Image Copy function. The IC or AIC command is required for each database that is being reorganized and is registered as RECOV in DBRC. If the ICPSYSIN DD statement is omitted, the Online Reorg function will automatically create an IMAGE COPY for all databases being reorganized that are registered as RECOV in DBRC. For more information, see “ICPSYSIN statements” on page 311.

**NOTE**

Although IMAGE COPY PLUS allows input from a SYSIN data set, the Image Copy function running during a reorganization requires input from ICPSYSIN. A SYSIN statement is ignored.

### output image copy DD

Required unless it is dynamically allocated (through keywords in the ICPSYSIN data set). Describes a data set to contain the output image copy created by the Image Copy function.

The IC keyword value in the ICPSYSIN data set identifies the ddname of this data set. One statement of this type is required for each explicitly allocated output image copy.

Do not specify DISP=(...,...,DELETE) for this data set. (This specification deletes the data set if the job step terminates abnormally.) If you are running multiple tasks per step and one of those tasks abends, the entire job step ends with the return code of the abended task. The other tasks in the job probably created good data sets (which the
Image Copy function recognizes as good data sets). However, if these data sets were specified with DELETE, they are all deleted if any task abends, even though the task that created the data set did not itself abend. BMC recommends that you use dynamic allocation or specify the data sets with DISP=(......,KEEP) or DISP=(......,CATLG).

BLKSIZE and BUFNO specifications on this DD statement might affect the determination of output block sizes and buffer pool sizes.

If you do not specify a block size on the DD statement, Image Copy function optimizes the block size for the device to which it writes the data. Therefore, as a general rule, omit the block size specification for best performance. If you do specify a block size and the data set is fixed-block (FB or FBS), the block size must be a multiple of the LRECL; otherwise, the Image Copy function PLUS uses the optimum block size for the device.

**PLUSDUMP DD**

Optional. This statement is a trigger to cause diagnostic information to be written to the PLUSOUT data set. If the PLUSOUT DD statement is not in the JCL, the output of PLUSDUMP is dynamically allocated as SYSOUT=*.

**NOTE**

This statement generates a large amount of output. Include it in your JCL only at the request of BMC Customer Support.

**REPORTS DD**

Optional. Defines the output data set that contains the Image Copy function statistical reports. Normally, this statement defines a SYSOUT data set. If you do not include a REPORTS DD statement, the reports are directed to the ICPPRINT DD data set.

The default DCB parameters are LRECL=121 and BLKSIZE=1210. To override these defaults, you must provide values for both parameters. However, Image Copy function only formats the first 121 positions of each record, even if you specify an LRECL greater than 121.

**ICPSYSIN statements**

ICPSYSIN statements use the same syntax as PLUSIN statements. They consist of a commands, keywords, separators, and continuation characters. For more information, see “PLUSIN control statement syntax” on page 404.
ICPSYSIN commands

You can specify the following commands for the Image Copy function. The command must be the first nonblank characters.

**NOTE**

You can use an existing ICPSYSIN data set from an IMAGE COPY PLUS job with the Image Copy function. Commands and keywords that do not apply to the Image Copy function are ignored.

Table 49 describes the valid commands.

Table 49 Image Copy function commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLBL</td>
<td>Optional. The GLBL control statement sets options that apply to all tasks in the current execution of the Image Copy function. You can use it to override options set in the global options module. Not all options can be overridden. If you supply a GLBL control statement, it must be the first statement in the ICPSYSIN data set. For more information, see “Global options modules” on page 318.</td>
</tr>
<tr>
<td>GROUP</td>
<td>Optional. Use the GROUP control statement to define a group of related image copy tasks and to set customization options that apply to the entire group. The group of related tasks includes all control statements that follow the GROUP control statement up to the end of the file or to any subsequent GROUP control statement. The Image Copy function generates an implied group with a default name of ICPSYSIN for any control statements that follow the optional GLBL control statement and precede the first specified GROUP control statement. The NAME or DBDSGRP keyword is required on the GROUP statement.</td>
</tr>
<tr>
<td>IC or AIC</td>
<td>Required. Use the IC or AIC control statement to invoke the batch image copy process for the specified data set group. The DBD keyword is required to identify the data set group to be copied. <strong>Note</strong>: For IMAGE COPY PLUS, the IC statement causes synchronous processing and the AIC statement causes asynchronous process. The Image Copy function processes the image copies asynchronously regardless of whether you use the IC or AIC statement.</td>
</tr>
</tbody>
</table>
ICPSYSIN keywords

Table 50 presents an overview of the options that you can set for the Image Copy function.

The control statement keywords enable and disable Image Copy function features, set values for specific tasks, and identify data sets.

The keywords are never positional; you can place them in any order on a control statement. If you do not specify a keyword, the Image Copy function uses the default value assigned for that keyword if one exists.

Table 50 lists the keywords that are valid on Image Copy function commands, accepted values, default values, the pages where you can find more information, and short descriptions of the use of the keywords.

### Table 50  Image Copy function ICPSYSIN keywords (part 1 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>GLBL</th>
<th>GROUP</th>
<th>IC/ AIC</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td></td>
<td></td>
<td>x</td>
<td>(dbdName)</td>
<td>none</td>
<td>420</td>
<td>Specify the DBD name for the primary DBD.</td>
</tr>
<tr>
<td>DBDSGRP</td>
<td></td>
<td>x</td>
<td></td>
<td>(dbdsName)</td>
<td>none</td>
<td>420</td>
<td>Generate image copy data sets for all data set groups defined in the database data set group.</td>
</tr>
<tr>
<td>DDN</td>
<td></td>
<td>x</td>
<td></td>
<td>(ddname)</td>
<td>none</td>
<td>423</td>
<td>Specify the ddname associated with the data set group to be processed.</td>
</tr>
<tr>
<td>DESC</td>
<td></td>
<td>x</td>
<td></td>
<td>(nn,...)</td>
<td>(7)</td>
<td>424</td>
<td>Specify the descriptor code numbers that designate the manner in which MVS is to manage the highlighting and retention of monitoring messages issued to MVS consoles.</td>
</tr>
<tr>
<td>HIST</td>
<td></td>
<td>x</td>
<td>x</td>
<td>(nnnn)</td>
<td>(9999)</td>
<td>447</td>
<td>Specify a limit for the generations of historical information that the Image Copy function maintains in the PDX for each execution of the Image Copy function.</td>
</tr>
<tr>
<td>IC</td>
<td></td>
<td>x</td>
<td>x</td>
<td>(ddname,ddname,...)</td>
<td>none</td>
<td>449</td>
<td>Specify the ddname of each output image copy data set to write.</td>
</tr>
</tbody>
</table>
### Table 50  Image Copy function ICPSYSIN keywords (part 2 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>GLBL</th>
<th>GROUP</th>
<th>IC/ AIC</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICACC</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(1) (2)</td>
<td>(1)</td>
<td>449</td>
<td>Specify the access method (1=SAM, 2=EXCP) to use for writing output image copy data sets.</td>
</tr>
<tr>
<td>ICALLOC</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>450</td>
<td>Control dynamic allocation of output image copy data sets if a DD statement is not included in the JCL.</td>
</tr>
<tr>
<td>ICAVGREC</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>() (U) (K) (M)</td>
<td>()</td>
<td>451</td>
<td>Specify the allocation units (blocks, logical records, or multiples of logical records) for DASD space when dynamically allocating new SMS output image copy data sets.</td>
</tr>
<tr>
<td>ICBUF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(0) (1) (2) (3) (4)</td>
<td>(4)</td>
<td>451</td>
<td>Specify the output image copy buffer option.</td>
</tr>
<tr>
<td>ICCATF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>452</td>
<td>Create catalog entries for dynamically allocated output image copies.</td>
</tr>
<tr>
<td>ICDATAACL</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>() (dataclas)</td>
<td>()</td>
<td>453</td>
<td>Specify the DATACLAS (data class) when you dynamically allocate new SMS image copy output data sets.</td>
</tr>
<tr>
<td>ICDSCB</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>() (dscbName)</td>
<td>none</td>
<td>454</td>
<td>Specify the name of the data set control block (DSCB) to use as a model for the generation data group (GDG) when you dynamically allocate the output image copy data sets.</td>
</tr>
<tr>
<td>ICEXPDT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(yyddd) (yyyyyddd)</td>
<td>none</td>
<td>454</td>
<td>Specify an expiration date for output image copy data sets when they are dynamically allocated.</td>
</tr>
<tr>
<td>ICGDG</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(N) (Y)</td>
<td>(N)</td>
<td>455</td>
<td>Create image copy data sets as GDGs when the output image copy data sets are dynamically allocated.</td>
</tr>
</tbody>
</table>
Table 50  Image Copy function ICPSYSIN keywords (part 3 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>GLBL</th>
<th>GROUP</th>
<th>IC/AIC</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICLIKE</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(): (dataSetName)</td>
<td>()</td>
<td>455</td>
<td>Specify the data set name of an existing SMS data set to use as a model when you dynamically allocate a new SMS image copy output data set.</td>
</tr>
<tr>
<td>ICMGMTCL</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(): (className)</td>
<td>()</td>
<td>456</td>
<td>Specify the name of the MGMTCLAS (management class) to use when you dynamically allocate new SMS image copy output data sets.</td>
</tr>
<tr>
<td>ICPREF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(): (dsnPrefix)</td>
<td>none</td>
<td>459</td>
<td>Identify a data set name prefix for the output image copy data set.</td>
</tr>
<tr>
<td>ICRETPD</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(): (value)</td>
<td>none</td>
<td>460</td>
<td>Specify a retention period (in days) for output image copy data sets that are dynamically allocated.</td>
</tr>
<tr>
<td>ICSPACE</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(): (unit,pri,sec)</td>
<td>()</td>
<td>461</td>
<td>Specify the amount of DASD space to allocate for a new non-SMS data set or to override the space allocation defined in the DATACLAS (data class) for an SMS data set.</td>
</tr>
<tr>
<td>ICSTORCL</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(): (className)</td>
<td>()</td>
<td>462</td>
<td>Specify the name of a STORCLAS (storage class) to use when you dynamically allocate new SMS image copy output data sets.</td>
</tr>
<tr>
<td>ICUNIT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(): (unitName,count)</td>
<td>(TAPE,1)</td>
<td>463</td>
<td>Specify the device unit name to use when dynamically allocating each output image copy data sets. Also specify the number of physical units to allocate.</td>
</tr>
<tr>
<td>ICVOLCNT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(): (value)</td>
<td>0</td>
<td>463</td>
<td>Override your installation’s MVS specification on the maximum number of volumes allowed for an output image copy data set when they are dynamically allocated.</td>
</tr>
</tbody>
</table>
### ICPSYSIN keywords

#### Table 50  Image Copy function ICPSYSIN keywords (part 4 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>GLBL</th>
<th>GROUP</th>
<th>IC/ AIC</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICVOLSER</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(volser,volser,...)</td>
<td>()</td>
<td>464</td>
<td>Override your installation’s MVS specification on the maximum number of volumes allowed for an output image copy data set when they are dynamically allocated.</td>
</tr>
<tr>
<td>INST</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(instName)</td>
<td>('BMC SOFTWARE, INC.')</td>
<td>476</td>
<td>Specify the name of your installation as you want it to appear in the heading portion of Image Copy function reports.</td>
</tr>
<tr>
<td>NAME</td>
<td></td>
<td></td>
<td></td>
<td>(grpName)</td>
<td>(CMDnnn)</td>
<td>493</td>
<td>Specify a unique identifier for a group that will be used by the Image Copy function.</td>
</tr>
<tr>
<td>PARMBLK</td>
<td>x</td>
<td></td>
<td></td>
<td>(modName)</td>
<td>(ICPPARMS)</td>
<td>507</td>
<td>Identify the name assigned to the global options module for the Image Copy function.</td>
</tr>
<tr>
<td>PCP</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(N) (Y)</td>
<td>(N)</td>
<td>517</td>
<td>Activate the interface to the Hash Checking technique of POINTER CHECKER PLUS® to verify the pointers in the database while taking an image copy of the database.</td>
</tr>
<tr>
<td>PDX</td>
<td>x</td>
<td></td>
<td></td>
<td>() (pdxName) (DUMMY) (NULLFILE)</td>
<td>()</td>
<td>518</td>
<td>Specify the name of the PDX data set that should be used to retrieve database specific options and to write processing statistics for the Image Copy function.</td>
</tr>
<tr>
<td>ROUTCDE</td>
<td>x</td>
<td></td>
<td></td>
<td>(nn,nn,...)</td>
<td>(1,7,11)</td>
<td>530</td>
<td>Specify a list of routing code numbers (1–15) that indicates which types of MVS operator consoles are to receive Image Copy function monitoring messages.</td>
</tr>
<tr>
<td>SIUDSN</td>
<td>x</td>
<td></td>
<td></td>
<td>(dsName)</td>
<td>(bmc.siudummy, dsn)</td>
<td>541</td>
<td>Specify a dummy data set name (1–44 characters) to use when notifying DBRC that a virtual image copy has been created.</td>
</tr>
</tbody>
</table>
User-controlled options

The Image Copy function allows you to set user-controlled options with different methods to suit your needs. The Image Copy function offers several levels for setting options (see Table 51). Each level overrides the options set in the previous level.

NOTE
If you already have IMAGE COPY PLUS, the Image Copy function can use any existing IMAGE COPY PLUS global options modules and database-specific options members. You do not need to recreate them.

Table 50  Image Copy function ICPSYSIN keywords (part 5 of 5)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>GLBL</th>
<th>GROUP</th>
<th>IC/ AIC</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATS</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>548</td>
<td>Specify whether to accumulate and print free space analysis information during Image Copy function processing.</td>
</tr>
<tr>
<td>TIME</td>
<td>x</td>
<td></td>
<td></td>
<td>(N) (Y)</td>
<td>(N)</td>
<td>556</td>
<td>Specify whether to provide a time stamp on Image Copy function messages in the processing log.</td>
</tr>
<tr>
<td>USERID</td>
<td>x</td>
<td></td>
<td></td>
<td>(user1,user2,...)</td>
<td>( )</td>
<td>573</td>
<td>Specify a list of one or more TSO users to notify when exceptional conditions occur during processing.</td>
</tr>
<tr>
<td>VIC</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(N) (Y)</td>
<td>(N)</td>
<td>573</td>
<td>Create a virtual image copy of the primary and secondary indexes for the reorganized database.</td>
</tr>
</tbody>
</table>

Table 51  Ways to specify options for the Image Copy function (part 1 of 2)

<table>
<thead>
<tr>
<th>Level</th>
<th>Used for</th>
<th>Resides in</th>
<th>Created by</th>
</tr>
</thead>
<tbody>
<tr>
<td>internal defaults</td>
<td>all jobs that don’t specify an options module or options in ICPSYSIN</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>ICPPARMS global options module</td>
<td>all jobs at your installation</td>
<td>load library</td>
<td>ISPF panels or #DBUGLBL batch job</td>
</tr>
<tr>
<td>ICP@imsi global options module</td>
<td>all jobs run on a particular IMS system (for example, a test system)</td>
<td>load library</td>
<td>ISPF panels or #DBUGLBL batch job</td>
</tr>
</tbody>
</table>
You can set up global options modules to specify the values you want to use for most databases most of the time. When it is available, the Image Copy function uses this module instead of the internal defaults.

You can create several global options modules, each with a different name, a different set of values, and a different purpose. For example, you can create one for your production system and another for your test system. If you have several IMS systems, you can have different sets of values for each system.

**ICPPARMS**

You can create a module named ICPPARMS to contain your installation default values. In the ICPPARMS module, change any values that your entire installation will use all of the time. Creating this module might address all your needs.

ICPPARMS is very important when you are creating database-specific options members with the ISPF interface. The Image Copy function initializes the ISPF panels with the values you specify in ICPPARMS. When ICPPARMS is not available, the Image Copy function initializes the panels with its internal default values.

**ICP@imsi**

You can create modules named ICP@imsi (where *imsi* is the IMSID) when you want an IMS system to have different default values from the global values. For example, you could create two ICP@imsi modules: one named ICP@TEST for the test system, and the other named ICP@PROD for the production system.
ICP@xxxx

You can create modules named ICP@xxxx (where xxxx is any four-character identifier) when you want to create other modules for other purposes. To use an ICP@xxxx module, specify the module name with the PARMBLK keyword in the ICPSYSIN data set.

Database-specific options members

You can set up database-specific options members in the PDX. The Image Copy function uses this member when you are performing an image copy process on the database associated with the member. A PDX is a special data set used by BMC’s Database Utilities products for storing customization options and reports. Details about PDX data sets can be found in the BMC IMS Database Supplemental Utilities Reference Manual.

You can create a different database-specific member for each database in your installation. You can also make these modules IMSID-specific or create one module for each database that will be used regardless of the IMSID. This allows you more control over the processing of each database.

Control statement keywords

For certain jobs, you might want to use a new set of values, so you could specify keywords on any of the Image Copy function control statements in the ICPSYSIN data set. The values for these keywords override the internal defaults and any values specified in the options modules. For more information, see “ICPSYSIN keywords” on page 313.

Image Copy function global options modules

You can create a global options module using either of two methods:

- batch
- ISPF panels

Batch method

If your installation does not use ISPF, or if you choose not to use the interface, you must use the batch method to create global options modules.
The ICPPARMS member in the sample library is a prototype for creating global options modules in batch. The keywords in the supplied source module match the system-defined keyword defaults.

You can modify the BMC-defined default values for the keywords in this module. After modifying the keyword values, reassemble and link the global options module using Assembler Level H. Assemble and link-edit the ICPPARMS member using the ICPASML member in the sample library. Modify the JCL to reflect the libraries for your installation. This global options module will not be used for jobs that already executing, but will be used for any jobs using this module that start after you link-edit the module into the library.

Figure 83 lists the macro keywords and their respective default values. You can modify the keywords as you wish.

---

**NOTE**

Some keywords in the ICPPARMS member do not apply to the Image Copy function in the MAXM Reorg solutions but are present for IMAGE COPY PLUS.

---

Figure 83   SAMPLIB member ICPPARMS (part 1 of 4)

```
* THE BMC-DEFINED DEFAULTS ARE CAREFULLY CHOSEN TO BE OPTIMAL. THESE DEFAULT VALUES CANNOT BE CHANGED, BUT YOU CAN OVERRIDE THEM WITH A SET OF CUSTOMER-DEFINED DEFAULT VALUES BY CREATING (AND MAINTAINING) A SET OF OPTIONS CALLED THE "ICPPARMS" OPTIONS MODULE.
* THE PREFERRED METHOD FOR CREATING A CUSTOMER-DEFINED OPTIONS MODULE IS TO USE THE BMC-SUPPLIED ISPF PANELS. IN CASES WHERE THIS IS INAPPROPRIATE, AN OPTIONS MODULE MAY BE CREATED USING THIS EXAMPLE ASSEMBLY LANGUAGE MODULE.
* NOTE THAT ASSEMBLER LEVEL H IS REQUIRED TO ASSEMBLE THIS MODULE.

***********************************************************************
ICPPARMS START 0
***********************************************************************
```

* THE CSECT NAME FOR THE OPTIONS MODULE MUST BE THE SAME AS THE LOAD MODULE (MEMBER) NAME. THE OPTIONS MODULE NAME MUST USE THE FOLLOWING CONVENTIONS:

* "ICPPARMS" - DEFAULT SYSTEM OPTIONS MODULE.
* "ICP@____" - ALTERNATE OPTIONS MODULE. THE "____" CHARACTERS MAY BE ANY 4 ALPHANUMERIC OR NATIONAL CHARACTERS (A-Z, 0-9, @#$).
* IN MANY CASES, THESE CHARACTERS ARE THE
Figure 83  SAMPLIB member ICPPARMS (part 2 of 4)

```plaintext
* "IMSID" OF A SPECIFIC SUBSYSTEM. *
*=====================================================================*

$RCUDEF   ID='**** DEFAULT PARAMETERS ****',
          PDX/TITLE/DBRC OPTIONS
  ASYNCH=N,                ASYNCHRONOUS SYSIN
  DBRC=Y,                  FORCE DBRC TO BE ACTIVE
  DESC=(7),                DESCRIPTOR CODES
  FLUSH=N,                 FLUSH ON ABEND
  GRPLIM=0,                GROUP WAIT LIMIT
  INST='BMC SOFTWARE, INC.', INSTALLATION NAME
  NOTIFY=Y,                NOTIFY DBRC OF IMAGE COPY
  PDX=,                    DEFAULT PDX DATA SET NAME
  ROUTCDE=(1,7,11),        ROUTING CODES
  DBACC=3,                 DATABASE ACCESS METHOD
  DBALLOC=Y,               DYNAM ALLOC DATABASES
  DBBUF=4,                 DATABASE I/O BUFFER OPT
  ICACC=2,                 OUTPUT IC ACCESS METHOD
  ICBUF=4,                 OUTPUT IC BUFFER OPT
  MERGE=Y,                 MERGE BY VOLUME COPIES
  VOLCOPY=N,               IMAGE COPY BY VOLUME
  ICALLOC=N,               DYNAM ALLOC OUTPUT IC
  ICPREF=,                 IMAGE COPY PREFIX
  ICUNIT=(TAPE,1),         OUTPUT IMAGE COPY UNIT
  ICVOLCNT=0,              OUTPUT VOLUME COUNT
  ICSTORCL=,               IC SMS STORCLAS
  ICVOLSER=,               IC VOLSER LIST
  FORCEIC=Y,               FORCE IC WHEN NOT REQ'D
  ICCATF=Y,                CATALOG STACKED IC'S
  ICDSGB=,                 MODEL DSCB
  ICEXPDT=,                EXPIRATION DATE
  ICGDG=N,                 OUTPUT IMAGE COPY GDG'S
  ICRETPD=,                RETENTION PERIOD
  ICSTORCL=,               IC SMS STORCLAS
  ICVOLSER=,               IC VOLSER LIST

Figure83 SAMPLIB member ICPPARMS (part2of4)
```
Figure 83  SAMPLIB member ICPPARMS (part 3 of 4)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VERIFY=Y</td>
<td>VERIFY INPUT TO CIC</td>
</tr>
<tr>
<td>ASIS=N</td>
<td>COPY AS IS OPTION</td>
</tr>
<tr>
<td>FSE=N</td>
<td>ZERO FREE SPACE ELEMENTS</td>
</tr>
<tr>
<td>ICCOMP=N</td>
<td>COMPRESS OUTPUT IC</td>
</tr>
<tr>
<td>ICALG=1</td>
<td>COMPRESSION ALGORITHM</td>
</tr>
<tr>
<td>FREQ=10000</td>
<td>LOG MONITOR MSGS ... RECS</td>
</tr>
<tr>
<td>HIST=9999</td>
<td>NUMBER OF HISTORY RECORDS</td>
</tr>
<tr>
<td>STATS=Y</td>
<td>PRINT FREE SPACE STATS</td>
</tr>
<tr>
<td>DBUSSID=DBUZ</td>
<td>DBU/SS IDENTIFIER</td>
</tr>
<tr>
<td>FPA=N</td>
<td>FAST PATH ANALYZER</td>
</tr>
<tr>
<td>MINCYL=0</td>
<td>MIN CYLS FOR DBACC(3)</td>
</tr>
<tr>
<td>PCP=N</td>
<td>POINTER CHECKER PLUS</td>
</tr>
<tr>
<td>SIUDSN=siudummy.dsn</td>
<td>VIRTUAL IMAGE COPY DSN</td>
</tr>
<tr>
<td>VIC=N</td>
<td>CREATE VIRTUAL IC’S</td>
</tr>
<tr>
<td>DSACC=2</td>
<td>ACCUM/LOG/IC ACCESS METHOD</td>
</tr>
<tr>
<td>DSALLOC=N</td>
<td>DYNAM ALLOC ACCUM/LOG/IC</td>
</tr>
<tr>
<td>DSBUF=4</td>
<td>ACCUM/LOG/IC BUFFER OPT</td>
</tr>
<tr>
<td>DSUNIT=(,1)</td>
<td>ACCUM/LOG/IC DSUNIT</td>
</tr>
<tr>
<td>AOI=N</td>
<td>ALLOW AOI?</td>
</tr>
<tr>
<td>AUTHINTV=2000</td>
<td>AUTH RETRY INTERVAL (SSHH)</td>
</tr>
<tr>
<td>CMDWAIT=15</td>
<td>AUTH RETRIES BEFORE CMD</td>
</tr>
<tr>
<td>DBR=N</td>
<td>AUTO /DBR DATABASE</td>
</tr>
<tr>
<td>FEOV=N</td>
<td>SWITCH IMS/VS SYS LOGS</td>
</tr>
<tr>
<td>STA=N</td>
<td>AUTO /STA DATABASE</td>
</tr>
<tr>
<td>AGN=</td>
<td>APPLICATION GROUP NAME</td>
</tr>
<tr>
<td>CICSID=</td>
<td>CICS-DL/I SYSTEM NAME</td>
</tr>
<tr>
<td>IMSID=</td>
<td>IMSID FOR AOI/BMP TASK</td>
</tr>
<tr>
<td>OPT=N</td>
<td>INACTIVE CNTL RGN OPTN</td>
</tr>
<tr>
<td>PSB=</td>
<td>PSB NAME FOR AOI/BMP TASK</td>
</tr>
<tr>
<td>TRAN=</td>
<td>TRAN NAME FOR AOI/BMP TASK</td>
</tr>
<tr>
<td>DBDLIB=</td>
<td>DEFAULT DBD LIBRARY NAME</td>
</tr>
</tbody>
</table>

Figure83 SAMPLIB member ICPPARMS (part 3 of 4)
ISPF method

If your installation uses ISPF, you can install the Database Utilities ISPF interface panels and use them to create global options modules. You can create the ICPPARMS, ICP@imsi, ICP@xxxx, and global options modules with the ISPF interface.

Perform the following steps to build global options modules using the ISPF interface:

1. Access the BMC Software IMS DBU Product Selection Menu.
2. Type S next to Backup and Recovery Products.
3. Press Enter.
4. Type 1 on the IMS Database Utilities - Backup and Recovery Products panel.
5. Press Enter. The IMAGE COPY PLUS primary menu is displayed.
6. Type 3 in the Selection field and press Enter. The Global Options panel is displayed.
7. The value 1 is currently displayed in the Selection field. Do not change this value. Press Return until the cursor is in the Options suffix field.
8. Type PARMS or any four-character options module suffix in the Options suffix field. The actual name assigned to the module is ICPPARMS or ICP@xxxx where xxxx is the four characters typed in the field. Press Enter. The PDX / Title / DBRC Options panel is displayed.
9. Change any or all values to suit the needs of your installation. Remember that these fields relate to the control statement keywords; the associated keyword is noted to the right of the field. For information about the keywords, see Chapter 14, “User-controlled options.”
10. Press Enter to move to the next panel. There are 14 panels in the Global Options series of panels. Continue to alter the values in the fields and press Enter until the Global Options panel is displayed.
Database-specific options modules

The Image Copy function uses a database-specific options member (if present) to obtain processing specifications for a database. You can create these members with the ISPF interface only. Perform the following procedure to create a database-specific options member in the PDX:

1. Access the BMC Software IMS DBU Product Selection Menu.
2. Type S next to Backup and Recovery Products.
3. Press Enter.
4. Type 1 on the IMS Database Utilities - Backup and Recovery Products panel.
5. Press Enter. The IMAGE COPY PLUS primary menu is displayed.
6. Type 2 in the Selection field and press Enter. The Database Selection List panel is displayed.
7. The DBD names listed on this panel come from the DBDLIB Type S in the S field next to the database for which you want to create a database-specific member and press Enter. The Database Specific Options panel is displayed.
8. The value 1 is currently displayed in the Selection field. Do not change this value. Press Enter. The Database I/O Options panel is displayed.
9. There are eight panels in the Database Specific Options series of panels. These panels are initialized with the values you specified for ICPPARMS, if available, or with the IMAGE COPY PLUS internal default values. Change any or all values to suit the needs of your installation. Remember that these fields relate to the control statement keywords; the associated keyword is noted to the right of the field. For information about the keywords, see Chapter 14, “User-controlled options.”
10. Press Enter to move to the next panel. Continue to alter the values in the fields and press Enter until the Database Specific Options panel is displayed.
11. Press F3, or the key designated as your END key. The IMAGE COPY PLUS primary menu appears. A message appears in the upper right corner informing you that the member has been saved.
Reports

The Image Copy function generates the reports shown in Table 52. For more information about the reports, refer to the page number shown.

<table>
<thead>
<tr>
<th>Report</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Summary</td>
<td>586</td>
</tr>
<tr>
<td>Data Set Group Summary</td>
<td>588</td>
</tr>
<tr>
<td>Global Specifications</td>
<td>596</td>
</tr>
<tr>
<td>Output Image Copy Summary</td>
<td>605</td>
</tr>
<tr>
<td>Processing Log</td>
<td>607</td>
</tr>
<tr>
<td>Processing Summary</td>
<td>608</td>
</tr>
</tbody>
</table>

Image copy data sets

When you use the Image Copy function, you must identify one or more output image copy data sets that the function will generate. To identify the output image copy data sets, code the IC keyword on the IC, AIC, or GROUP control statement. If you specify the keyword on the GROUP control statement, it is optional on the IC or AIC control statement.

You can explicitly code DD statements for the output image copy data sets in the JCL, or you can have the Image Copy function dynamically allocate the data sets.

Explicit allocation of image copy data sets

To allocate output image copy data set explicitly, perform the following tasks:

- Code the IC keyword in the ICPSYSIN data set. As the keyword value, specify one or more ddnames (one for each output image copy you want to generate). Each ddname refers to a DD statement in the JCL and must be unique within the execution. Code a separator between the ddnames.

- Code a DD statement for each output image copy data set. For more information, see “JCL requirements” on page 310.
For example, the following statements explicitly allocate output image copy data sets using DD statements named ICOUT1 and ICOUT2.

```
//ICOUT1 DD DSN=ACCOUNTS.PAYROLL1.ICOUT1.
//                   DISP=(,CATLG,CATLG),UNIT=TAPE
//ICOUT2 DD DSN=ACCOUNTS.PAYROLL1.ICOUT2.
//                   DISP=(,CATLG,CATLG),UNIT=TAPE

//ICPSYSIN DD *
IC DBD(PAYROLL1) DDN(PYRLDSG1) -
  IC(ICOUT1,ICOUT2) ....
```

When a DD statement is present in the JCL, Image Copy function ignores dynamic allocation keywords coded on control statements and in global options modules and database-specific options members. You must supply all pertinent information with appropriate JCL parameters on the DD statement, such as EXPDT, RETPD, UNIT, and VOL.

### Dynamic allocation of image copy data sets

To allocate output image copy data set dynamically, perform the following tasks:

- Code the IC keyword in the ICPSYSIN data set. As the keyword value, specify one or more ddnames (one for each output image copy you want to generate), or specify one or more asterisks (*), one for each output image copy.
- Omit the output image copy data set DD statements from the execution JCL.
- Code ICALLOC(Y) in a valid control statement or options module. If you coded IC(*), you can omit ICALLOC(Y).
- You can also code the following keywords in a valid control statement or options module to control the dynamic allocation of the output data set.
  - ICAVGREC
  - ICCATF
  - ICDATAACL
  - ICDSCB
  - ICEXPDT
  - ICGDG
  - ICLIKE
  - ICMGMTCL
  - ICPIREF
  - ICETPD
Examples

The following example dynamically allocates all output image copy data sets for three databases:

```
//ICPSYSIN DD *
  IC DBD(PAYROLL1) DDN(PYRLDSG1) IC(*) ...
  IC DBD(BENEFITS) AREA(CLASS1) IC(*) ...
  IC DBD(FINANCE1) AREA(FINAREA1) IC(*) ...
```

The following example performs the same processing.

```
//ICPSYSIN DD *
  GROUP NAME(GROUPA) IC(*) ...
  IC DBD(PAYROLL1) DDN(PYRLDSG1) ...
  IC DBD(BENEFITS) AREA(CLASS1) ...
  IC DBD(FINANCE1) AREA(FINAREA1) ...
```

The following example dynamically allocates multiple image copy data sets for three databases.

```
//ICPSYSIN DD *
  IC DBD(PAYROLL1) DDN(PYRLDSG1) IC(*,*,*) ...
  IC DBD(BENEFITS) AREA(CLASS1) IC(*,*,*) ...
  IC DBD(FINANCE1) AREA(FINAREA1) IC(*,*,*) ...
```

Allocation on a tape device

When you write the output image copy data set to a tape device using dynamic allocation, you must specify a tape device name using the ICUNIT keyword. You can code the following keywords to control the additional allocation and disposition options:

- ICCATF
- ICDSCB
- ICEXPDT
- ICGDG
- ICRETPD
Dynamic allocation of image copy data sets

- ICVOLCNT
- ICVOLSER

If the output image copy data set might exceed one tape volume (ICVOLSER or ICVOLCNT keywords), consider allocating multiple tape devices using the ICUNIT keyword. This can reduce the time delay required to mount the additional volumes. However, the Image Copy function allocates a maximum of two tape devices.

Do not include the following keywords when allocating the output image copy data set to a tape device:

- ICAVGREC
- ICDATAACL
- ICLIKE
- ICMGMTCL
- ICSPACE
- ICSTORCL

**Allocation on DASD**

When you write the output image copy data set to non-Storage Management Subsystem (SMS) DASD using dynamic allocation, you must specify a DASD device name using the ICUNIT keyword. You must also specify the amount of space to allocate using the ICSPACE keyword. BMC recommends that you code ICCATF(Y) to create catalog entries for the output data sets. You can include the following keywords to control the additional allocation options:

- ICCATF
- ICDSCB
- ICEXPDT
- ICGDG
- ICRETPD
- ICVOLSER
- ICVOLCNT

Do not include the following keywords when allocating the output image copy to a non-SMS DASD device:

- ICAVGREC
- ICDATAACL
- ICLIKE
- ICMGMTCL
- ICSTORCL
Do not specify multiple DASD devices using the ICUNIT keyword. If the data set might require space on multiple DASD volumes, code multiple volume serial IDs using the ICVOLSER keyword.

**Allocation using SMS**

When you write the output image copy to an SMS data set using dynamic allocation, you must specify one or more of the SMS-specific keywords or ensure that the data set meets the criteria established by your user-written automatic class selection (ACS) routine. The SMS-specific keywords are as follows:

- ICAVGREC
- ICDATAACL
- ICLIKE
- ICMGMTCL
- ICSPACE
- ICSTORCL

You can specify the following keywords to control the additional allocation options:

- ICCATF
- ICDSCB
- ICEXPDT
- ICGDG
- ICRETPD
- ICSPACE
- ICUNIT
- ICVOLCNT
- ICVOLSER

**Data set name construction**

When dynamically allocating non-stacked output image copies, the data set name will be constructed, using internal data set naming conventions, as follows:

```
prefix.ICn.dbdName.dsg
```

- The variables are defined as follows:
  - *prefix* is the value found in ICPREF keyword
  - *ICn*, where *n* = copies desired, values 1 through 10
  - *dbdName* is the DBD name
  - *dsg* is the ddname of the data set group or area
When the data set is a generation data group, as specified with the ICGDG(Y) keyword, the Image Copy function concatenates the GDG generation number, and the data set name is in the following format:

```
prefix.1Ch.dbdName.dsg(+1)
```

**WARNING**

If GDGs are preallocated, do not specify SYS, PLUS, and DFS prefixes. For performance reasons, ddnames with these prefixes cannot be used; they will be ignored.

You can control the name that is assigned to the dynamically allocated data set by using the Data Set Name user exit (ICP$DSN). For details, see "Using the image copy Data Set Name user exit" on page 678.

If the Image Copy function finds an existing, cataloged data set that matches the generated name, the new image copy overwrites the existing data set. If an existing data set is not found, the new image copy is written to a new dynamically allocated data set.

The Image Copy function can dynamically deallocate the output image copy data set when processing completes.

**CATDS versus NOCATDS**

DBRC provides an option for using the system catalog to manage certain characteristics about image copy data sets. Refer to the appropriate IMS Utilities reference manual for details about the CATDS and NOCATDS parameters for the INIT.RECON command.

When you initialize the RECON data set by using the NOCATDS parameter, DBRC records the unit and volume in the RECON data sets. Any information in the system catalog is ignored.

When you initialize the RECON data set by using the CATDS parameter, DBRC allows the unit and volume information recorded in the system catalog to override values recorded in the RECON data set. For this reason, it is important that image copy data sets are cataloged when you specify the CATDS parameter.

**Virtual image copies**

A virtual image copy is a data set in name only; it does not physically exist. The data set name is recorded in DBRC as an image copy of an index database, but an actual image copy data set does not exist.
Virtual image copies are useful in an environment where you plan to rebuild primary and secondary index databases rather than recovering them. In such an environment, you do not need to create a physical image copy data set for the index databases because they will not be used for recovery purposes.

The image copies defined to DBRC must be assigned the NOREUS parameter.

To allow DBRC to use appropriate purge times for change accumulation processing, BMC recommends that you perform a NOTIFY.IC function for the index databases. When you create a virtual image copy, the Image Copy function performs this notification automatically.

The Image Copy function supports creating an image copy of a shared secondary index data set if, on the IC control statement, you specify the first shared secondary index by using the DBD keyword. The Image Copy function does not support virtual image copies for shared secondary indexes.

To create a virtual image copy, perform the following tasks:

- Specify the VIC keyword in a valid control statement or options module.
- Specify the SIUDSN keyword in a valid control statement or options module.
- Use dynamic allocation members for the data set groups, or provide data set group DD statements.

Output image copy write options

You can control several options for writing output image copy data sets.

Access method control

The Image Copy function has two access methods available to write the image copy data sets.

- Native access method (SAM)

This method writes the data set at the same speed as an optimally tuned IMS Database Image Copy utility. You can still take advantage of the other Image Copy function features when you use this access method. To use SAM, specify 1 for the ICACC keyword.
EXCP-level access methods

This method is available for processing tape data sets only. It writes the data set at speeds that approach the physical capabilities of the tape device. To use EXCP, specify 2 for the ICACC keyword.

**NOTE**
The EXCP-level access method achieves better performance. BMC recommends that you use it in all cases.

Buffer pool size specification

You can manually specify the number of buffers for data sets using either access method. The number of buffers you specify affects the performance of the Image Copy function. Generally, the more buffers you use, the better the performance. If you have a heavy paging environment, fewer buffers might give you better performance.

When using SAM, the Image Copy function allocates all buffers in 24-bit storage, which is below the 16 MB line. Since buffers are allocated for each data set, there is a limit to the number of data sets that can be opened for concurrent processing. In particular, when you are creating dual (or multiple) copies of an image copy data set using SAM, separate buffers are allocated to each copy.

When using the EXCP method, the Image Copy function allocates all buffers in 31-bit storage, which is above the 16 MB line for MVS/XA and MVS/ESA systems. This greatly increases the number of data sets that can be opened for concurrent processing. In addition, when you create multiple copies of an image copy data set using EXCP, the Image Copy function allocates only one set of buffers regardless of the number of copies being created.

IMAGE COPY PLUS can dynamically allocate an optimum number of output buffers for image copy data sets based on the physical device and block size used. You can also explicitly specify the number of buffers in the JCL.

To control the number of buffers allocated, code the ICBUF keyword. IMAGE COPY PLUS calculates the number of buffers to allocate using the value specified with the ICBUF keyword. If you code DCB=BUFNO=n in the output image copy data set DD statement, and if this value is larger than the value calculated using the ICBUF keyword, this value overrides the ICBUF keyword.
Image copy block size settings

The Image Copy function can dynamically (implicitly) select the optimum block size when writing an image copy data set based on the physical device and database characteristics. When using the EXCP access method to write the image copy, the Image Copy function always selects the optimum block size. When using the SAM access method to write the image copy, you can explicitly specify the image copy data set block size.

If you explicitly code a valid block size on the output image copy data set DD statement, the Image Copy function uses that block size. If you omit the block size from the DD statement, or if you explicitly code an invalid one, the Image Copy function implicitly selects the optimum block size.

Explicit method

If you specify ICACC(1), explicitly identify the image copy block size by coding the following on the output image copy data set DD statement in the JCL:

\[ DCB=BLKSIZE=nnn \]

You must select a block size that is a multiple of the logical record length for the image copy data set. This is equal to the database block size (if OSAM) or control interval size (if VSAM) of the data set group or area being processed plus eight bytes.

**NOTE**

For KSDS and HISAM overflow data sets, the image copy logical record length is calculated using the logical record length of the database data set group instead of the control interval size or block size.

The minimum logical record length is 64 bytes.

Implicit method

If you allow the Image Copy function to determine the block size for the image copy data set, it selects the optimum value for the output device type that matches the rules outlined for the explicit method.

For tape devices, the block size is as large as possible, usually around 32 KB. For disk devices, the block size is selected based on the track capacity to minimize the amount of storage space required.
POINTER CHECKER PLUS interface

POINTER CHECKER PLUS® verifies the validity of both logical and physical pointers in an IMS database. It functionally replaces the IMS/VS DB Analyzer and the IMS/VS System Utilities/Database Tools HD Pointer Checker utility, which was previously available with the Space Management Utilities II (SMU-II).

After you have generated an image copy, you will usually want to verify both the logical and the physical pointers in the database. If you have POINTER CHECKER PLUS installed, you can check the pointers concurrently with the image copy process, thus saving time and system resources.

Option values are passed to the POINTER CHECKER PLUS interface in the same way as if POINTER CHECKER PLUS were running independently. Both a POINTER CHECKER PLUS options module and a PCPSYSIN DD statement can be used.

NOTE
POINTER CHECKER PLUS space monitoring functions are not performed when running under IMAGE COPY PLUS.

The POINTER CHECKER PLUS keyword RETCODE(\textit{nnnn}) is used to specify the return code passed back to the Image Copy function when the hash checking technique discovers an error.

Perform the following tasks to turn on the POINTER CHECKER PLUS interface:

- Specify the PCP keyword in a valid control statement or options module.
- Specify a PCPSYSIN DD statement. Refer to the POINTER CHECKER PLUS Reference Manual for instructions.

NOTE
If a database contains direct logical pointers, the database is not valid until prefix resolution is completed. Therefore, ICP(YES) is not valid for databases with direct logical pointers.

Monitoring and reporting options

You can control monitoring and reporting information produced by the Image Copy function. Most installations choose to set these options in their ICPPARMS global options module. You can, however, override the default values by coding the keywords on control statements in the ICPSYSIN data set.
Monitor the progress of the job

You can use the FREQ keyword to control output of Image Copy function monitoring information about the job. You can control how often the Image Copy function writes a message to the processing log or system console about the number of records it has read or written. At the end of record processing, the Image Copy function reports the total number of records processed.

For example, to have the Image Copy function write a message after reading or writing each 2000 records (2000, 4000, 6000, and so forth) until all records have been processed, specify FREQ(2000). To have no messages written, specify FREQ(0).

Free space analysis information

You can use the STATS keyword to control whether the Image Copy function will accumulate and print free space analysis information. Collection of the space analysis data increases the CPU time by about .01 CPU second per megabyte of data.

PDX history limit settings

You can use the HIST keyword to specify a limit on the generations of historical information that the Image Copy function maintains for each image copy function. Each generation requires 512 bytes in the PDX. The more generations you maintain, the more time is required to run the function.

For example, to maintain historical information about the 50 most recent Image Copy functions processed, specify HIST(50). To maintain the existing historical information in the PDX, but do not include information from this execution, specify HIST(0).

GROUP control statement

The GROUP control statement allows you to specify keywords that will be in effect for all functions within the group. If you code a keyword in the GROUP control statement, it is like placing the keyword in every control statement within the GROUP. If the same keyword appears on a function control statement, the Image Copy function uses the value in the function control statement.
User-defined GROUP processing

In the following example, the Image Copy function is requested to dynamically allocate most data sets.

```
//ICPSYSIN DD *
GROUP NAME(GROUPA)
 IC DBD(ABCD123) ICALLOC(Y)
 IC DBD(ABCD234) ICALLOC(Y)
 IC DBD(ABCD345) ICALLOC(N)
 AIC DBD(ABCD567) ICALLOC(Y)
```

You can include ICALLOC(Y) in the GROUP control statement to achieve the same results.

```
//ICPSYSIN DD *
GROUP NAME(GROUPA) ICALLOC(Y)
 IC DBD(ABCD123)
 IC DBD(ABCD234)
 IC DBD(ABCD345) ICALLOC(N)
 AIC DBD(ABCD567)
```

If you omit the GROUP control statement, the Image Copy function generates a GROUP statement for you.

The Image Copy function allows you to select from two types of GROUP processing:

- user-defined GROUP processing
- DBRC-defined GROUP processing

A user-defined group consists of a GROUP control statement followed by one or more function control statements. These function control statements identify the data set groups or areas to be processed, and the function to be performed.

A DBRC-defined group consists of a GROUP control statement that references a list of data set groups and areas registered within DBRC. The GROUP control statement, in this case, identifies the data set groups or areas to be processed and the function to be performed. The GROUP control statement can be followed by one or more function control statements.

**User-defined GROUP processing**

Perform the following tasks to define GROUP processing:

- Code a GROUP control statement that contains the NAME keyword.
- Code any another keywords in the GROUP control statement that should be used for each IC or AIC function that is part of the GROUP.
Code one or more IC or AIC control statements. Keyword values that you code on the IC or AIC control statement override any values coded in the GROUP control statement, the GLBL control statement, and the global options module.

Code the DBD keyword on the IC or AIC control statement to identify the database to process.

If necessary, code the DDN keyword on the IC or AIC control statement to identify the data set group to process. The value of the keyword is the ddname associated with a data set group in the database definition.

If you omit this keyword, the Image Copy function implicitly processes each data set group defined in the database.

### DBRC-defined GROUP processing

Perform the following tasks to enable DBRC-defined GROUP processing:

- Code a GROUP control statement that contains the DBDSSGRP keyword. The value of the keyword must match a database data set group (DBDSSGRP) that you have previously registered with DBRC. The Image Copy function implicitly processes all of the data set groups and areas defined to DBRC as members of the change accumulation group or database data set group.

- Code the FUNC keyword in the GROUP control statement to identify the Image Copy function (IC or AIC) that should be performed to all data sets listed in the DBRC-defined GROUP. Code any another keywords in the GROUP control statement that the Image Copy function should use for each function that is part of the GROUP.

- If necessary, you can code one or more IC or AIC control statements to identify additional data set groups and areas to be processed or to override values coded in the GROUP control statement for a data set group included implicitly from the DBRC-defined group.

- Code the DBD keyword on a control statement to identify the database to process.

- If necessary, code the DDN keyword on a control statement to identify the data set group to process. The value of the keyword is the ddname associated with a data set group in the database definition.

If you omit this keyword, the Image Copy function implicitly processes each data set group defined in the database. If the database contains only one data set group, you can omit the DDN keyword.
RECON Substitution Facility

This chapter describes how to use the RECON Substitution Facility. This chapter contains the following topics:

Overview ................................................................. 339
RSF Extract function .................................................. 340
RSF Extract keywords .................................................. 341
RSF execution intercept .............................................. 342
Creating a test database .............................................. 343

Overview

The BMC RECON Substitution Facility (RSF) restores simplicity to the test or development environment. RSF creates high availability large databases (HALDBs) that use a pseudo-RECON to provide more flexibility in your development and test environments. Because you can create or change definitions at every level of testing or development, you can control your data and database structures.

RSF extracts information from existing RECONs and intercepts DLI applications or utilities that are running. RSF creates HALDBs with a pseudo-RECON and generates all of the required Database Recovery Control (DBRC) INIT commands. When you move the HALDB into production, you can import the pseudo-RECON information to the actual production RECONS. You can also use RSF with standard full-function databases.

To use RSF, you need to identify the HALDB and specify standard DBRC INIT.DB and INIT.PART control statements in a BMCRECON DD statement. You can create these statements in the following ways:

- manually, by entering them in a standard 80-byte PDS member or sequential data set
RSF Extract function

- automatically, by using the MAXM Database Advisor for IMS HALDB Modeling tool
- automatically, by using the RSF RECON Extract function

RSF Extract function

To invoke the RECON Extract function, execute PGM=DLIGENTR. Specify READRCN and optional keywords in the PLUSIN control statement. You can direct the output to the RCNOUT DD statement or a sequential or PDS data set with LRECL=80.

The following example shows sample JCL for the extract process:

```
//BMCRCFEX JOB (acct),CLASS=C,MSGCLASS=X,MSGLEVEL=(1,1)
//DLIGENTR EXEC PGM=DLIGENTR
//STEPLIB DD DISP=SHR,DSN=BMCDLI.LOADLIB
// DD DISP=SHR,DSN=IMSVS.MDALIB
// IMS DD DISP=SHR,DSN=DBU.QA.ALL.DBDLIB
//BMCTRACE DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 
//RECON1 DD DISP=SHR,DSN=IMSVS.RECON1
//RECON2 DD DISP=SHR,DSN=IMSVS.RECON2
//RECON3 DD DISP=SHR,DSN=IMSVS.RECON3
//RCNOUT DD DISP=SHR,DSN=BMCDLI.RECOUT.PDS(MEMBER)
//RCNOUT DD DISP=SHR,DSN=BMCDLI.RECOUT.MEMBER
//RCNOUT DD DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,LRECL=80,
// PLUSIN DD *
//READRCN DBD(ANYDBD1) SSID(IMSP) INITRCN(Y) -
//DSNPREFIX(BMCDLI.NEWDSN.&DBDNAME.&PARTNAME)
/*
RSF Extract keywords

All extract keywords are optional. The following table describes the extract keywords.

Table 53  RSF Extract keywords

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Values allowed</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD(name1, name2, name3, name4)</td>
<td>up to four DBD names</td>
<td>This keyword specifies the DBD names and RECONs for the data extraction.</td>
</tr>
<tr>
<td></td>
<td>You can use a wildcard mask (*) to represent additional DBDs.</td>
<td>DBD(A*) extracts any DBD that has a name beginning with the letter A.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DBD(A,B,Z*) extracts databases A and B and any DBD starting with the letter Z.</td>
</tr>
<tr>
<td>DSNPREFX(hlq)</td>
<td>one high-level qualifier (up to 37 characters)</td>
<td>This keyword is valid only for HALDBs. It changes the DSNPREFX used for HALDBs during the RECON extract. It also includes the prefix in the DBRC INIT commands that are generated.</td>
</tr>
<tr>
<td>FFDSN(hlq)</td>
<td>one high-level qualifier (up to 44 characters)</td>
<td>This keyword is valid only for full-function databases. It change the data set name that is used for full-function databases during the extract. It also uses the specified name in the DBRC INIT commands that are generated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can make additional substitutions by inserting the following variable values that are gathered from the existing DBD:</td>
</tr>
</tbody>
</table>
|           |                                     |   - &DBDNAME  
|           |                                     |   - &DDNAME  
|           |                                     |   - &PARTNAME  
| SSID(name) | one name (up to 8 characters)       | This keyword specifies the IMSID to be used for an INIT.RECON command.      |
|           |                                     | The default is BMC1.                                                         |
| INITRCN(Y | N)      | Y or N                               | Specify Y to include an INIT.RECON command in the resulting DBRC commands that are extracted. |
RSF execution intercept

You can execute any DLI batch application or database reorganization utility (BMC or native IMS) by using RSF against any database (HALDB or non-HALDB). DBRC commands that are issued during these executions are not retained for further processing, and existing RECON data sets are not altered. DBRC authorization and interaction is only simulated.

To invoke RSF, the BMC product LOADLIB is required in STEPLIB and must reside before the IMS RESLIB, as the following example shows:

```
//USERPGMO EXEC PGM=DFSRRCO0,
//     PARM=(DLI,&PGMNAME,&PSBNAME,........
//        &IMSID,,&DBRC,&IRLM,........,&IMSPLEX)
//STEPLIB DD DISP=SHR,DSN=BMCDLI.LOADLIB
// DD DISP=SHR,DSN=HLQ.PGMLIB
// DD DISP=SHR,DSN=IMSVS.RESLIB

//BMCRECON DD DISP=SHR,DSN=BMCDLI.RECOUT.PDS(MEMBER)

or

//BMCRECON DD *
INIT.RECON SSID(IMSP)
INIT.DB
INIT.PART
INIT.DBDS

or

//BMCRECON DD DISP=SHR,DSN=BMCDLI.RCNOUT.DSN
```
Creating a test database

With RSF, you can create databases using BMC Reorg utilities or standard DLI, as if the databases were registered to DBRC.

You can clone databases by using the MAXM Reorg/EP or MAXM Reorg/Online Copy function. The following example copies all partitions, Indirect List Datasets (ILDSs), and PSINDEXs that are related to the selected HALDB.

```
//MXOCOPY EXEC PGM=DLICPYON
.
//PLUSIN DD *
COPY DBD(XIUODBD) SWAP(N) DBRC(Y) IDCAMS(MODEL) -
   DYNALLOC(Y,Y,C) SECINDEX(YES)
```

The copied data sets are modeled after the original data sets, and .C is appended to the original DSNPREFX as defined by the DYNALLOC keyword.

The RSF extract is then used to create the INIT commands for the new name, as shown in the following example:

```
//DLIGENTR EXEC PGM=DLIGENTR
.
//PLUSIN DD *
READRCN DBRC(Y) DBD(XIUO*) DSNPREFX(ORIG.HLQ.&DBDNAME.C)
//RCNOUT DD DISP=SHR,DSN=BMCDLI.RECOUT.PDS(MEMBER)
```
The following example shows output from the extract for a HALDB with two PSINDEXes:

<table>
<thead>
<tr>
<th>INIT.DB</th>
<th>INIT.PART</th>
<th>INIT.PART</th>
<th>INIT.PART</th>
<th>INIT.PART</th>
</tr>
</thead>
<tbody>
<tr>
<td>INIT.DB</td>
<td>DBD(XIU00DBD) TYPHALDB SHARELVL(3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INIT.PART</td>
<td>DBD(XIU00DBD) PART(XIU0A) - DNSPREFIX(ORIG.HLQ.XIU00DBD.C) - KEYSRNG(X'F3F7F1FF00000000') - BLOCKSZE(8192) - FBFF(30) - FSPF(25)</td>
<td>DBD(XIU00DBD) PART(XIU0B) - DNSPREFIX(ORIG.HLQ.XIU00DBD.C) - KEYSRNG(X'F7F6F7FF00000000') - FSPF(10)</td>
<td>DBD(XIU00DBD) PART(XIU0C) - DNSPREFIX(ORIG.HLQ.XIU00DBD.C) - KEYSRNG(X'F000000000000000') - FSPF(15)</td>
<td></td>
</tr>
<tr>
<td>INIT.PART</td>
<td>DBD(XIU0NDX1) TYPHALDB SHARELVL(3)</td>
<td>DBD(XIU0NDX1) PART(XIU0X1A) - DNSPREFIX(ORIG.HLQ.XIU0NDX1.C) - KEYSRNG(X'FFFFFFFF000000000000') - FSPF(10)</td>
<td>DBD(XIU0NDX2) TYPHALDB SHARELVL(3)</td>
<td>DBD(XIU0NDX2) PART(XIU0X2A) - DNSPREFIX(ORIG.HLQ.XIU0NDX2.C) - KEYSRNG(X'FFFFFFFF000000000000') - FSPF(10)</td>
</tr>
</tbody>
</table>

Subsequent DLI executions use the BMCRECON DD to access the copied data sets.

```sql
//BMCRECON DD DISP=SHR,DSN=BMCDLI.RECOUT.PDS(MEMBER)
```
# JCL requirements

This chapter describes the required, optional, and recommended JCL statements that you need to complete a database reorganization. It also shows JCL requirements. This chapter includes the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>346</td>
</tr>
<tr>
<td>EXEC statement</td>
<td>347</td>
</tr>
<tr>
<td>DD statements</td>
<td>349</td>
</tr>
<tr>
<td>AMSOUT DD</td>
<td>351</td>
</tr>
<tr>
<td>AMSPDS DD</td>
<td>352</td>
</tr>
<tr>
<td>ANLPKEYO DD</td>
<td>353</td>
</tr>
<tr>
<td>ANLPRINT DD</td>
<td>353</td>
</tr>
<tr>
<td>BMCMSG DD</td>
<td>353</td>
</tr>
<tr>
<td>BMCPRINT DD</td>
<td>354</td>
</tr>
<tr>
<td>BMCSETUP DD</td>
<td>355</td>
</tr>
<tr>
<td>BMCTRACE DD</td>
<td>355</td>
</tr>
<tr>
<td>CRFRDER/CRFRDER2 DD</td>
<td>356</td>
</tr>
<tr>
<td>DFSRESLB DD</td>
<td>357</td>
</tr>
<tr>
<td>DFSUINnn DD</td>
<td>358</td>
</tr>
<tr>
<td>DFSUINPT DD</td>
<td>358</td>
</tr>
<tr>
<td>DFSURCDS DD</td>
<td>359</td>
</tr>
<tr>
<td>DFSURGU1 DD</td>
<td>360</td>
</tr>
<tr>
<td>DFSURGU2 DD</td>
<td>360</td>
</tr>
<tr>
<td>DFSURGUn DD</td>
<td>361</td>
</tr>
<tr>
<td>DFSURGnn DD</td>
<td>362</td>
</tr>
<tr>
<td>DFSURIDX DD</td>
<td>363</td>
</tr>
<tr>
<td>DFSURWF1 DD</td>
<td>364</td>
</tr>
<tr>
<td>DFSURWF2 DD</td>
<td>365</td>
</tr>
<tr>
<td>DFSURWF3 DD</td>
<td>366</td>
</tr>
<tr>
<td>DMNCNTLI DD</td>
<td>367</td>
</tr>
<tr>
<td>EXTRACTO DD</td>
<td>367</td>
</tr>
<tr>
<td>HLDURIDX DD</td>
<td>367</td>
</tr>
<tr>
<td>HSRCNTLI DD</td>
<td>368</td>
</tr>
<tr>
<td>ICPPRINT DD</td>
<td>368</td>
</tr>
<tr>
<td>IFRDER/IEFRDER2 DD</td>
<td>369</td>
</tr>
<tr>
<td>IMS DD</td>
<td>370</td>
</tr>
<tr>
<td>OLRKEYS DD</td>
<td>370</td>
</tr>
</tbody>
</table>
The REORG command, available in the Online Reorg function and the Reorg function, calls and coordinates the reorganization. For a fast and automated reorganization, use the REORG command. If you need to perform only one or two reorganization tasks, you can run the Unload function, the Reload function, the Index Build function, or the Prefix Resolution/Update function as stand-alone utilities.

The JCL requirements for the MAXM Reorg solutions are similar to the JCL requirements for the classic database utilities, MAXM Reorg classic, and the IMS reorganization utilities. You can use existing reorganization JCL, with a few changes. Ensure that the BMC load library is ahead of the RESLIB in the STEPLIB concatenation.

The following DD statements are required for MAXM Reorg solutions:

- **STEPLIB**
- **DFSRESLIB**
- **IMS** (or DBDLIB. The DBDLIB DD statement is allowed for compatibility with SECONDARY INDEX UTILITY.)

All other DD statements are optional. Many data sets can be dynamically allocated. Print data sets (BMCPRINT, BMCMSG, and so on) are dynamically allocated if they are not specified in the JCL. Database data sets and output image copy data sets can be dynamically allocated.
BMC recommends that you do not specify DCB information for output files. MAXM Reorg solutions use system-determined block size to determine the DCB information needed for a file.

Some functions require additional DD statements. For more information, see Table 56 on page 350 and the chapters that describe the individual functions.

## EXEC statement

The EXEC statement for each utility except the Index Build function and the Prefix Resolution/Update function uses the same format as the EXEC statement that is used for the IMS utilities. It calls the region controller DFSRRC00 and specifies a ULU region. DBRC parameters can be included on the EXEC statement. You can add the IMSID parameter to override the IMSID specified in the IMS RESLIB.

The MAXM Reorg solutions do not run as an IMS dependent region.

### Table 54 EXEC statements by command (part 1 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>EXEC statement</th>
</tr>
</thead>
</table>
| REORG (Online Reorg function and Reorg function) | PGM=DFSRRC00,PARM='ULU,DFSURGL0,dbname'  
where dbname is the name of the database being reorganized. |
| UNLOAD (Unload function)      | PGM=DFSRRC00,PARM='ULU,DFSURGU0,dbname'  
where dbname is the name of the database being unloaded |
| RELOAD (Reload function)      | PGM=DFSRRC00,PARM='ULU,DFSURGL0,dbname'  
where dbname is the name of the database being reloaded |
| BUILD/BUILDALL (Index Build function) | PGM=BMCSIU |
The JCL requirements are similar for all MAXM Reorg solutions. The functions use the commands shown in Table 55.

### Table 54  EXEC statements by command (part 2 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>EXEC statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefix Resolution/Update function</td>
<td>- For the IMS-compatible Prefix Resolution function and High-performance Prefix Resolution function: PGM=DFSURG10 or PGM=PRPURG10</td>
</tr>
<tr>
<td></td>
<td>- For the Two-step Prefix Resolution function:</td>
</tr>
<tr>
<td></td>
<td>- Parent-Child Resolution step: PGM=PRPURG20</td>
</tr>
<tr>
<td></td>
<td>- Logical Twin Resolution step: PGM=PRPURG30</td>
</tr>
<tr>
<td></td>
<td>- For the Stand-alone Prefix Update function: PGM=PRPURGP0</td>
</tr>
<tr>
<td></td>
<td>- For the Concurrent Prefix Update function: PGM=DFSURG10</td>
</tr>
<tr>
<td></td>
<td>- For the Secondary Index Resolution function: PGM=PRPURG40</td>
</tr>
<tr>
<td></td>
<td>- For the Work File 1 Generator function: PGM=DFSURGC00,PARM='ULU,DFSURG0,dbname'</td>
</tr>
<tr>
<td></td>
<td>- For the Work File 1 Splitter function: PGM=PRPSPLIT</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>The Image Copy function runs under the Online Reorg function and Reorg function only; it cannot run as a stand-alone utility.</td>
</tr>
</tbody>
</table>

### Table 55  Commands (part 1 of 2)

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online Reorg function</td>
<td>REORG</td>
<td>Reorganize a database while updates are being made by an online IMS or CICS/DBCTL system by using the Reorg function, the Unload function, the Reload function, the Index Build function, and the Image Copy function.</td>
</tr>
<tr>
<td>Reorg function</td>
<td>REORG</td>
<td>Reorganize a database by using the Unload function, the Reload function, and the Index Build function. Prefix resolution and update is also integrated for logical relationships.</td>
</tr>
<tr>
<td>Unload function</td>
<td>UNLOAD</td>
<td>Unload the records in a database.</td>
</tr>
<tr>
<td>Unload function</td>
<td>DBSELECT</td>
<td>Retrieve selected records from a database. For more information, see “Database Selection facility” on page 192.</td>
</tr>
</tbody>
</table>
DD statements

This section describes the required, optional, and recommended DD statements (see Table 56 on page 350) and shows sample JCL for each product. The DD statements are described in alphabetical order. Each description provides the purpose of the statement.

*NOTE*

The DD statements are described in general. If additional explanation is necessary for a particular command, the explanation is included after the general description.

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unload function</td>
<td>TRACE</td>
<td>Use the API. For more information, see “API” on page 199.</td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>USERREAD</td>
<td></td>
</tr>
<tr>
<td>Reload function</td>
<td>RELOAD</td>
<td>Reload the records in a database.</td>
</tr>
<tr>
<td>Reload function</td>
<td>SORT</td>
<td>Sort unloaded records before reloading them by using the stand-alone HD Sort utility. For more information, see “HD Sort utility” on page 227.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To sort and reload a database, specify the HDSORT keyword with the RELOAD command.</td>
</tr>
<tr>
<td>Reload function</td>
<td>USERLOAD</td>
<td>Use the API to load a database initially. For more information, see “API” on page 234.</td>
</tr>
<tr>
<td>Index Build function</td>
<td>BUILD</td>
<td>Build indexes (primary and/or secondary) for a database.</td>
</tr>
<tr>
<td></td>
<td>BUILDALL*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XFLD*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XSCN*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XLOD*</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>*These commands are converted to a BUILD command automatically.</td>
</tr>
<tr>
<td>Index Build function</td>
<td>SPLIT</td>
<td>Split the WF1 file and write index records to one file and logical relationship records to another file.</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>IC</td>
<td>Create output image copies during the database reorganization process. Keywords are specified in the ICPSYSIN data set.</td>
</tr>
<tr>
<td></td>
<td>AIC</td>
<td></td>
</tr>
<tr>
<td>Image Copy function</td>
<td>GLBL</td>
<td>Set global and group options for the Image Copy function. Keywords are specified in the ICPSYSIN data set.</td>
</tr>
<tr>
<td></td>
<td>GROUP</td>
<td></td>
</tr>
</tbody>
</table>
### Table 56 DD statements (part 1 of 2)

<table>
<thead>
<tr>
<th>Statement</th>
<th>REORG (Online Reorg function)</th>
<th>REORG (Reorg function)</th>
<th>UNLOAD</th>
<th>RELOAD</th>
<th>Index Build function</th>
<th>Prefix Resolution/Update function</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMSOUT</td>
<td>required</td>
<td>recommended</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>AMSPDS</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>optional</td>
</tr>
<tr>
<td>BMCMSG</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>n/a</td>
</tr>
<tr>
<td>BMCPRINT</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>n/a</td>
</tr>
<tr>
<td>BMCSETUP</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>n/a</td>
</tr>
<tr>
<td>BMCTRACE</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>n/a</td>
</tr>
<tr>
<td>CRFRDER</td>
<td>required</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>CRFRDER2</td>
<td>optional</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSRESLB</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSUINnn</td>
<td>n/a</td>
<td>optional</td>
<td>n/a</td>
<td>required if DFSUINPT is not provided</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSUINPT</td>
<td>n/a</td>
<td>optional</td>
<td>n/a</td>
<td>required if DFSUINnn is not provided</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSURCDS</td>
<td>required for logical relationships</td>
<td>required for logical relationships</td>
<td>n/a</td>
<td>required for logical relationships</td>
<td>n/a</td>
<td>required</td>
</tr>
<tr>
<td>DFSURGU1</td>
<td>n/a</td>
<td>n/a</td>
<td>required</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSURGU2</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSURGU3n</td>
<td>n/a</td>
<td>n/a</td>
<td>optional</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSURGnn</td>
<td>n/a</td>
<td>n/a</td>
<td>optional</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSURIDX</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>required to sort secondary index records</td>
<td>n/a</td>
</tr>
<tr>
<td>DFSURWF1</td>
<td>n/a</td>
<td>optional for logical relationships</td>
<td>n/a</td>
<td>required for logical relationships or secondary indexes</td>
<td>see page 364</td>
<td>required if no SORTIN DD statement provided</td>
</tr>
<tr>
<td>DFSURWF2</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>required</td>
</tr>
<tr>
<td>DFSURWF3</td>
<td>n/a</td>
<td>optional for logical relationships</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>optional for IMS-compatible Prefix Resolution; required for Stand-alone Prefix Update</td>
</tr>
<tr>
<td>HLDURIDX</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>optional</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>
The data set named in the AMSOUT DD statement contains the restart information that is required for the following tasks:

- restart post-processing tasks that failed
- swap data set names after the reorganization when SWAP(NO) or SWAP(S,JC) was specified on the REORG command

### Table 56 DD statements (part 2 of 2)

<table>
<thead>
<tr>
<th>Statement</th>
<th>REORG (Online Reorg function)</th>
<th>REORG (Reorg function)</th>
<th>UNLOAD</th>
<th>RELOAD</th>
<th>Index Build function</th>
<th>Prefix Resolution/Update function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICPSYSIN</td>
<td>required</td>
<td>required if ICP(Y) is specified</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>IEFRRDER2</td>
<td>optional</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>IMS</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>recommended</td>
</tr>
<tr>
<td>PLUSIN</td>
<td>optional</td>
<td>required</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td>PLUSLIST</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>optimal</td>
</tr>
<tr>
<td>PLUSOUT</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>optional</td>
</tr>
<tr>
<td>PRPURIDX</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>optional</td>
</tr>
<tr>
<td>PRPURWF1</td>
<td>n/a</td>
<td>optional for logical relationships</td>
<td>n/a</td>
<td>optional</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>print</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>n/a</td>
</tr>
<tr>
<td>RECON</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
<td>optional</td>
</tr>
<tr>
<td>SEQERROR</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>optional</td>
<td>n/a</td>
<td>required</td>
</tr>
<tr>
<td>SORTIN</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>SORTWK</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>required</td>
</tr>
<tr>
<td>STEPLIB</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>required</td>
<td>required</td>
</tr>
<tr>
<td>SYSOUT</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>required</td>
</tr>
<tr>
<td>ICPPRINT</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>required</td>
</tr>
<tr>
<td>SYSUDUMP</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>optional</td>
</tr>
</tbody>
</table>
AMSPDS DD

NOTE
Do not modify the AMSOUT file.

Table 57 describes the AMSOUT DD requirements by command.

Table 57  AMSOUT DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Required. You can specify a sequential file or a partitioned data set (PDS or PDSE). BMC recommends a PDSE. If you choose a PDS or PDSE, the member name is the name of the primary database. You can define this file in the Reorg function global options module. The file has a record length of 80 bytes. For more information, see “Post-processing phase” on page 125.</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Recommended. You can specify a sequential file, a PDS, or a PDSE. BMC recommends a PDSE. If you choose a PDS or PDSE, the member name is the name of the primary database. You can define this file in the Reorg function global options module. The file has a record length of 80 bytes. For more information, see “Post-processing phase” on page 103.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

AMSPDS DD

The data set named in the AMSPDS DD statement contains IDCAMS DELETE/DEFINE statements. Table 58 describes the AMSPDS DD requirements by command.

Table 58  AMSPDS DD requirements by command (part 1 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Optional</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Optional</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional</td>
</tr>
</tbody>
</table>
ANLPKEYO DD

Defines the output data set from Online/Defrag Analyzer. Contains the list of keys identifying the records that are recommended for reorganization. The ANLPKEYO DD statement is renamed OLRKEYS when processed by the Online/Defrag job.

The Online/Defrag Analyzer requires the ANLPKEYO DD statement.

ANLPRINT DD

Defines the data sets that contain reports generated by the Online/Defrag Analyzer. The Online/Defrag Analyzer requires the ANLPRINT DD statement.

BMCMSG DD

The data set named in the BMCMSG DD statement defines the output data set containing any messages issued by the utilities and a listing of PLUSIN control statements. Normally this statement defines a SYSOUT data set. The default parameters are LRECL=121 and BLKSIZE=1210. If the BMCMSG data set does not exist, the utility dynamically allocates it.

Table 59 on page 354 describes the BMCMSG DD requirements by command.

NOTE

When you specify IDCAMS(*) for REORG (both), RELOAD, and BUILD, specify a PDS that contains IDCAMS DELETE/DEFINE statements:

- For BUILD and RELOAD, specify a data set that contains DELETE/DEFINE statements the real data sets.
- For REORG and COPY, specify a data set that contains DELETE/DEFINE statements the shadow data sets.

Table 58  AMSPDS DD requirements by command (part 2 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUILD</td>
<td>Optional</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>Optional</td>
</tr>
</tbody>
</table>
The data set named in the BMCPRINT DD statement defines the output data set containing reports. Normally, this statement defines a SYSOUT data set. The default parameters are LRECL=121 and BLKSIZE=1210. You can specify a different data set for reports by using the xxxPRINT keyword. For more information, see “xxxPRINT” on page 522.

The utility searches for a data set you specified with xxxPRINT. If you did not specify a data set with xxxPRINT, the utility sends the reports to the BMCPRINT data set. If the BMCPRINT data set does not exist, the utility dynamically allocates it.

Table 60 describes the BMCPRINT DD requirements by command.

Table 60  BMCPRINT DD requirements by command (part 1 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Optional</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Optional</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>BUILD</td>
<td>Optional</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

NOTE

When performing multiple database reorganizations or copies, BMCMSG cannot be allocated on a PDSE.
BMCSETUP DD

The BMCSETUP DD statement describes a control statement data set that allows you to override the setup options that were set during installation. Table 61 describes the BMCSETUP DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Optional</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Optional</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>BUILD</td>
<td>Optional</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

**NOTE**

When performing multiple database reorganizations or copies, BMCPRINT cannot be allocated on a PDSE.

BMCTRACE DD

The data set named in the BMCTRACE DD statement defines the data set containing information that might be needed by BMC Customer Support to analyze a problem. This data set is dynamically allocated if any of the functions abends. However, include a BMCTRACE DD statement if you need to analyze the performance of the job. For more information, see “BMCTRACE DD” on page 621.
Table 62 describes the BMCTRACE DD requirements by command.

### Table 62  BMCTRACE DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Optional</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Optional</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>BUILD</td>
<td>Optional</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

**NOTE**

When performing multiple database reorganizations or copies, BMCTRACE cannot be allocated on a PDSE.

---

**CRFRDER/CRFRDER2 DD**

The data set named in the CRFRDER DD statement describes the log data set created during the log recover process. If you specify the LOGDSN keyword, the Online Reorg function dynamically allocates this data set.

The CRFRDER/CRFRDER2 DD statements and IEFRDER/IEFRDER2 DD statements must be paired. You can pair the two DD statements by using dynamic allocation or by specifying it in the JCL.

The CRFRDER data set contains modified information from the IEFRDER data set. The Online Reorg function creates the CRFRDER data set to provide the log records required for a recovery (if a recovery is needed). The CRFRDER data set is automatically registered to the RECON data set as a batch log. If you define a new data set, you must specify DISP=(NEW,CATLG,CATLG).

The data set named in the CRFRDER2 DD statement describes the log data set created during the log recover process if you use dual logging. If you specify LOGDSN(\textit{dsnMask},DUAL), Online Reorg function dynamically allocates this data set. If your installation requires dual logging, specify CRFRDER and CRFRDER2. If you define a new data set, you must specify DISP=(NEW,CATLG,CATLG).
Table 63 describes the CRFRDER DD requirements by command.

### Table 63  CRFRDER DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Required; can be dynamically allocated if you specify the LOGDSN keyword</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

**DFSSRESLB DD**

The data set named in the DFSRESLB DD statement describes the library containing the RESLIB.

Table 64 describes the DFSRESLB DD requirements by command.

### Table 64  DFSRESLB DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Required</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Required</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Required</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Required</td>
</tr>
<tr>
<td>BUILD</td>
<td>Required</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>Required</td>
</tr>
</tbody>
</table>
DFSUINnn DD

The data set named in the DFSUINnn DD statement describes the input data set containing the data to be reloaded. This data set must have been created by the Unload function. This data set is used as input to the Reload function or to the Reorg function (when you specify INPUT(UNLDFILE)). If you have multiple HD Unload data sets that were created by a parallel unload task, you can specify them on multiple DFSUINnn DD statements regardless of whether you reload them in parallel mode. For information about how to specify the DFSUINnn DD statement, see Table 215 on page 511.

Table 65 describes the DFSUINnn DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Required if you specify INPUT(UNLDFILE) and no DFSUINPT DD statement is provided.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Required if no DFSUINPT DD statement is provided. For details, see Table 215 on page 511.</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

DFSUINPT DD

The data set named in the DFSUINPT DD statement describes the unload file that will be used as input to a non-parallel reload. This is the data set created by the IMS HD Reorganization Unload utility (DFSURGU1 DD statement), the Unload function, or any other program that produces a data set in the HD Unload format. This data set is used as input to the Reload function or to the Reorg function (when you specify INPUT(UNLDFILE)).

If you unloaded the database in parallel mode, you can reload in non-parallel mode by concatenating the unload data sets in the correct sequence using a DFSUINPT DD statement.

If you specify a DFSUINPT DD statement, a DFSUINnn DD statement is not allowed.

Table 66 on page 359 describes the DFSUINPT DD requirements by command.
The data set named in the DFSURCDS DD statement describes the control data set containing the data generated by the IMS Prereorganization utility. If you specify DUMMY for the DFSURWF1 DD statement, you do not need the DFSURCDS DD statement.

For multiple database reorganizations, all logically related databases must be in the same DFSURCDS.

Table 67 describes the DFSURCDS DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Required if there are logical relationships.</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Required if there are logical relationships.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Required if there are logical relationships.</td>
</tr>
<tr>
<td>BUILD</td>
<td>Required if there are logical relationships.</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Required. Describes the control data set containing the data generated by the IMS Prereorganization utility.</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>
**DFSURGU1 DD**

The data set named in the DFSURGU1 DD statement describes the output data set that will contain the unloaded data when you unload the database in non-parallel mode. Table 68 describes the DFSURGU1 DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Required. The Unload function calculates the optimum block size, based on the device type, in the same way as the system-determined block size in DFSMS. You can override the Unload function block size by specifying the DCB=BLKSIZE parameter. If you want the Unload function to read the database to verify all pointers and generate the various reports without writing the unload records, specify DD DUMMY.</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

**DFSURGU2 DD**

The data set named in the DFSURGU2 DD varies, depending on which command is used. Table 69 describes the DFSURGU2 DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Optional. If you include a DFSURGU2 DD statement that does not specify DUMMY, the Online Reorg function writes the HD Unload file to the DFSURGU2 data set rather than to temporary data sets. If the DFSURGU2 data set is on DASD, you must provide sufficient disk space specification in the JCL. Use the DFSURGU2 DD statement to override the space allocations (device, number of cylinders, and so on) that were set in the environment setup member (DLIGSET0). To change only the number of cylinders for the HD Unload file, specify the number of cylinders for the CRFUNSP option in the environment setup member.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The data sets named in the DFSURGUn DD statements describe the output data sets used when you unload the database in parallel mode. DFSURGUn is the default ddname. (You can use different ddnames by specifying them with the LIMITS keyword.)

### Table 69  DFSURG2 DD requirements by command (part 2 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Reorg function)</td>
<td>Optional. If you want the Reorg function to create an HD Unload file, specify a DFSURG2 DD statement. The Reorg function does not use the HD Unload file, and creating one degrades performance.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Optional. If you want two copies of the unload data set, you must specify PARALLEL(NO). DFSURG2 contains the copy data set for the unload. The data set can reside on tape or a DASD device. The discussion of the DCB parameter in the DFSURG1 DD statement (see page 360) applies to this DD statement. The LIMITS keyword is not valid when you specify PARALLEL(NO). If you are creating two unload copies on tape and you specified the SYNCTAPE keyword in the UNLOAD command, any BLKSIZE and BUFNO values specified on the DFSURG2 DD statement are ignored. If you are unloading the database in parallel mode, DFSURG2 is the default ddname of the second subset of the unloaded database. (You can use different ddnames by specifying them with the LIMITS keyword.) To unload a database in parallel mode, you must specify PARALLEL(YES), or accept it as the default, and specify the LIMITS keyword. For more information, see “LIMITS” on page 479. <strong>Note:</strong> If you unload a database in parallel mode, you cannot request a second copy of the unload file.</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>
Table 70 describes the DFSURGUn DD requirements by command.

Table 70  DFSURGUn DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Optional. When you specify PARALLEL(YES) and LIMITS (except LIMITS(PART)), you can specify up to five output data sets. You must specify sequentially numbered data sets (1, 2, and so on). These data sets can reside on tape or a DASD device. The discussion of the DCB parameter in the DFSURGU1 DD statement (see page 360) applies to this DD statement.</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

DFSURGnn DD

The data sets named in the DFSURGnn DD statements describe the output data sets used when you unload a database that uses the IMS/ESA Partition Support Product and you specify LIMITS(PART).

Table 71 describes the DFSURGUnn DD requirements by command.

Table 71  DFSURGUnn DD requirements by command (part 1 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
</tbody>
</table>
The data set named in the DFSURIDX DD statement describes the data set containing the input index records.

Table 72 describes the DFSURIDX DD requirements by command.

Table 72  DFSURIDX DD requirements by command (part 1 of 2)

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>Required if you specify INDD(DFSURIDX) with the BUILDALL command.</td>
</tr>
</tbody>
</table>
The data set named in the DFSURWF1 DD statement defines the work data set to be used as output from the Reload function and as input to the secondary index creation and prefix resolution process. This data set contains IMS-format logical relationship and secondary index records. You do not need to specify DCB information. You can override the block size by specifying the DCB=BLKSIZE parameter. The RECFM=VB and LRECL=900 parameters are the defaults; you cannot override them. If you do not specify a block size, the system-determined block size is used.

Table 73 on page 365 describes the DFSURWF1 DD requirements by command.
The data set named in the DFSURWF2 DD statement defines a temporary work data set that Prefix Resolution/Update function creates during the Parent-Child Resolution phase (PRPURG20). Prefix Resolution/Update function uses the DFSURWF2 data set as input to the Logical Twin Resolution phase (PRPURG30).

### Table 73  DFSURWF1 DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>The DFSURWF1 DD statement may not be specified. The Online Reorg function is fully integrated with the Index Build function and the prefix resolution and update function in the Reorg function to perform secondary index and prefix resolution tasks.</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Omit the DFSURWF1 and PRPURWF1 DD statements to perform secondary index and prefix resolution tasks automatically by the Reorg function. If you do not want the Online Reorg function to build secondary indexes, you can specify DUMMY for the DFSURWF1 DD statement.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Required if there are logical relationships or secondary indexes. Defines the work data set created during the reload that is used as input to the secondary index creation and prefix resolution processes. If the records that are placed in this data set are not needed, you can specify DD DUMMY. The Reload function will bypass the generation of these records, saving CPU cycles. To write logical relationship records to PRPURWF1, do not specify DD DUMMY.</td>
</tr>
<tr>
<td>BUILD</td>
<td>Required if you specify the INDD keyword. When you specify the INDD keyword, the DFSURWF1 DD statement names the input data set for the index build process.</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Required if you do not supply a SORTIN DD statement. This statement indicates that the input to prefix resolution contains only IMS-format logical relationship and secondary index records. If the IBM Work File 1 Generator was used when this data set was created, you must code the DCB information on the DD statement.</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

### DFSURWF2 DD

The data set named in the DFSURWF2 DD statement defines a temporary work data set that Prefix Resolution/Update function creates during the Parent-Child Resolution phase (PRPURG20). Prefix Resolution/Update function uses the DFSURWF2 data set as input to the Logical Twin Resolution phase (PRPURG30).
**NOTE**
For easier restart, create this data set with a disposition that will not cause its deletion until the Logical Twin Resolution phase completes. You can use DFSURWF2 as input to PRPURG30 to restart the Logical Twin Resolution phase if DFSURG10 abends after completion of the Parent-Child Resolution phase.

Table 74 describes the DFSURWF2 DD requirements by command.

### Table 74  DFSURWF2 DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Required</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

**DFSURWF3 DD**

The DFSURWF3 DD statement is used to collect prefix update records when the prefix resolution and update function is integrated with the Reorg function execution. You do not need to specify DCB information. If DFSURWF3 is specified, prefix resolution will be performed but prefix update will be deferred and must be performed in a separate step.

**NOTE**
If you are using the GROUP subcommand to reorganize multiple databases simultaneously, DFSURWF3 is ignored.

Table 75 on page 367 describes the DFSURWF3 DD requirements by command.
Chapter 13 JCL requirements

**DMNCNTLI DD**

The data set named in the DMNCNTLI DD statement contains an 80-byte sequential file that contains the recommended control cards for the Online/Defrag Analyzer (DLIMDMAN). You may specify DMNCNTLI without modification as the PLUSIN DD statement for the Online/Defrag Analyzer.

The Online/Defrag Analyzer requires the DMNCNTLI DD statement.

**EXTRACTO DD**

Defines the data set extracted by Online/Defrag.

Online/Defrag requires the EXTRACTO DD statement for record mode reorganizations.

**HLDURIDX DD**

The data set named in the HLDURIDX DD statement describes the data set that contains secondary records that are not in the DFSURWF1 data set if you split secondary index records.

Table 76 on page 368 describes the HLDURIDX DD requirements by command.

---

Table 75  DFSURWF3 DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Optional. If you specify UPDATE(Y), omit this data set.</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

---
The data set name in the HSRCNTLI DD statement contains an 80-byte sequential file that contains the recommended control cards for Online/Defrag (BMCOLR). The HSRCNTLI DD statement must be edited to include the SSID keyword. The alternative is to concatenate the HSRCNTLI DD statement with other keywords in subsequent data sets.

Online/Defrag requires the HSRCNTLI DD statement.

ICPPRINT DD

The data set named in the ICPPRINT DD statement defines the output data set containing the reports and any error messages for logical relationship processing and image copy processing. Normally it defines a SYSOUT data set.

Table 77 describes the ICPPRINT DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional. If you include an HLDURIDX DD statement and do not include an OUTDDX keyword, all secondary index WFI records are written to the data set specified for HLDURIDX.</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Table 77 ICPPRINT DD requirements by command (part 1 of 2)
The data set named in the IEFRDER DD statement describes the log data set created during the change apply process. If you specify the LOGDSN keyword, the Online Reorg function dynamically allocates this data set. If you define a new data set, you must specify DISP=(NEW,CATLG,CATLG). If you do not supply DCB information, the Online Reorg function uses 30 buffers and a block size equal to 32 KB (for tape) or half track (for disk).

The data set named in the IEFRDER2 DD statement describes the dual log data set created during the change apply process if you use dual logging. If you specify LOGDSN(dsnMask,DUAL), the Online Reorg function dynamically allocates this data set. If your installation requires dual logging, specify IEFRDER and IEFRDER2. If you define a new data set, you must specify DISP=(NEW,CATLG,CATLG). If you do not supply DCB information, the Online Reorg function uses 30 buffers and a block size equal to 32 KB (for tape) or half track (for disk).

Table 78 describes the IEFRDER and IEFRDER2 DD requirements by command.
The data set named in the IMS DD statement describes the library, typically IMSVS.DBDLIB, containing the DBD referenced in the EXEC statement PARM operand.

Table 79 describes the IMS DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

### OLRKEYS DD

The OLRKEYS DD statement describes the data set that contains the record keys which Online/Defrag Analyzer recommends for reorganization. The record length can be variable or fixed, but it must be large enough to hold the root key. The OLRKEYS data set can be created by the Online/Defrag analysis.

The OREORG function of Online/Defrag Analyzer requires the HSRCNTLI DD statement when you specify MODE(RECORD) and you do not specify the KEYS keyword.
PCPDANIP DD

Defines the input data set for Online/Defrag Analyzer.

Online/Defrag Analyzer requires the PCPDANIP DD statement.

PDX DD

The data set named in the PDX DD statement defines the PDX data set containing output reports and statistics. You can explicitly bypass the use of a PDX data set in the following ways:

- Specify PDX DD DUMMY in the JCL.
- Specify the PDX data set name as blanks in the global options module and do not include a PDX DD statement in the JCL. The global options module can be named xxxPARMS (where xxx is the product code) or xxx@imsi (where imsi is the IMSID). Because global options modules are created with the classic database utilities rather than the MAXM Reorg solution the following product codes are valid:
  - ULP
  - LDP
  - SIU
  - PRP
  - ICP

  If you specify blanks in the xxxPARMS and/or xxx@imsi global options module but you include a PDX DD statement in the JCL, the utility uses the PDX.

To use a PDX data set, perform either of the following actions:

- Specify PDX DD DSN=pdxName in the JCL.
- Specify a PDX in the xxxPARMS and/or xxx@imsi global options module.

The utility searches the JCL first. If it finds a PDX DD statement, the PDX is used. If the PDX DD statement is not in the JCL, the utility uses the PDX entry from the xxx@imsi global options module. If neither is found, the utility uses the PDX entry from the xxxPARMS module.

Table 80 on page 372 describes the PDX DD requirements by command.
### PLUSIN DD

The PLUSIN DD statement describes the control statement data set containing the command and optional keywords for the functions to use during the reorganization. The record length must be 80. “PLUSIN control statement syntax” on page 404 describes how to code statements in the PLUSIN data set. Chapter 14, “User-controlled options” explains the valid keywords and values.

Table 81 on page 373 describes the PLUSIN DD requirements by command.

#### Table 80  PDX DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function and Reorg function)</td>
<td>Optional. If you specify a PDX DD statement when you use the REORG command, the reports generated by the Unload function, the Reload function, and the Index Build function are stored in the specified data set. The reports generated by the Online Reorg function and the Reorg function are not stored in the PDX data set; they are written to the BMCPRINT data set or the data set specified with the FRFPRINT keyword.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Optional. If you want to bypass the PDX but do not want to use this statement and you do not want the Unload function to set a return code of 04, specify the PDX data set name as blanks in the ULPPARMS or the ULP@imsi global options module.</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional. If you want to bypass the PDX but do not want to use this statement and you do not want the Reload function to set a return code of 04, specify the PDX data set name as blanks in the LDPPARMS or the LDP@imsi global options module.</td>
</tr>
<tr>
<td>BUILD</td>
<td>Optional. If you want to bypass the PDX but do not want to use this statement and you do not want the Index Build function to set a return code of 04, specify the PDX data set name as blanks in the SIUPARMS or the SIU@imsi global options module.</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Optional. If you do not specify this statement, the Prefix Resolution/Update function attempts to use the PDX you specify with the PDX option. To bypass use of the PDX, specify PDX DD DUMMY.</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>Optional. If you do not specify this statement, the Image Copy function attempts to use the PDX you specify with the PDX keyword in the ICPSYSIN data set or Image Copy function global options module. To bypass use of the PDX, specify PDX DD DUMMY or set the PDX keyword to DUMMY or NULLFILE.</td>
</tr>
</tbody>
</table>

**NOTE**

PDX is not valid with HALDBs.
Table 81  PLUSIN DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>Required. The REORG command is required. The SHARE(YES,UPDATE) and keyword is required.</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>Required. The REORG command is required.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>BUILD</td>
<td>Required. The BUILD command and the DBD keyword are required.</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Optional. If you do not supply this DD statement, the Prefix Resolution/Update function dynamically allocates it to the default SYSOUT class (MSGCLASS).</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

PLUSLIST DD

The PLUSLIST DD statement defines the output data set that contains a listing of all control statements and any error messages. Normally it defines a SYSOUT data set.

Table 82 describes the PLUSLIST DD requirements by command.

Table 82  PLUSLIST DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Optional. If you do not supply this DD statement, the Prefix Resolution/Update function dynamically allocates it to the default SYSOUT class (MSGCLASS).</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

PLUSOUT DD

The PLUSOUT DD statement defines the output data set that contains snaps of miscellaneous control blocks used for debugging. Normally it defines a SYSOUT data set.

Table 83 on page 374 describes the PLUSOUT DD requirements by command.
The ddname named in the xxxPRINT keyword defines the output data set for reports generated by all functions (except the Prefix Resolution/Update function and the Image Copy function). The reports are written to the data set defined for that utility. For more information, see “xxxPRINT” on page 522.

Table 83 describes the Print DD requirements by command.

### Table 83  PLUSOUT DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Optional</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>Optional; if you omit this statement, the Image Copy function dynamically allocates PLUSOUT to the default SYSOUT class (MSGCLASS).</td>
</tr>
</tbody>
</table>

### print DD

The ddname named in the xxxPRINT keyword defines the output data set for reports generated by all functions (except the Prefix Resolution/Update function and the Image Copy function). The reports are written to the data set defined for that utility. For more information, see “xxxPRINT” on page 522.

Table 84 describes the Print DD requirements by command.

### Table 84  Print DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function and Reorg function)</td>
<td>Optional. You can specify the FRFPRINT, ULPPRINT, LDPPRINT, and/or SIUPRINT keywords with the REORG command. When you specify these keywords, you can supply corresponding DD statements. When you use MAXM Reorg/Online or specify ICP(YES) in the Reorg function, the Image Copy function is invoked. You can specify an ICPPRINT DD statement to specify a data set that will contain the reports generated by the Image Copy function. If you do not specify the ICPPRINT DD statement, an ICPPRINT file is dynamically allocated.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional</td>
</tr>
<tr>
<td>BUILD</td>
<td>Optional</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>
PRPURIDX DD

The data set named in the PRPURIDX data set defines the work data set that the Prefix Resolution/Update function creates during its Secondary Index Resolution program (which it invokes when you SORTIDX(Y)). PRPURIDX is a temporary work data set. If you do not specify this DD statement and you specify SORTIDX(Y), the Prefix Resolution/Update function dynamically allocates this data set to the unit name you specify with the UNIT option.

Table 85 describes the PRPURIDX DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>LOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Optional</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

PRPURWF1 DD

The data set named in the PRPURWF1 DD statement defines the work data set to be used as input to the prefix resolution and update processes.

You must include a DFSURWF1 DD statement with the PRPURWF1 DD statement even though the data set allocated in the DFSURWF1 DD statement is not used.

You do not need to specify DCB information. You can override the block size by specifying the DCB=BLKSIZE parameter. The RECFM=VB and LRECL=900 parameters are the defaults; you cannot override them. If you do not specify a block size, the system-determined block size is used.

Table 86 on page 376 describes the PRPURWF1 DD requirements by command.
RECONx DD

The data sets named in the RECONx DD statements define the DBRC RECON data sets. The RECONx DD statements are required for HALDBs. The RECONx DD statements are optional for full-function databases. BMC strongly recommends that you do not include them in the JCL for full-function databases. Instead, add an MDALIB that contains the RECON data sets to your STEPLIB concatenation. RECON MDAs can be stored in the RESLIB.

Table 87 describes the RECONx DD requirements by command.

**Table 87  RECONx DD requirements by command**

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function)</td>
<td>The PRPURWF1 DD statement may not be specified. The Online Reorg function is fully integrated with the Index Build function and the prefix resolution and update function to perform secondary index and prefix resolution tasks.</td>
</tr>
<tr>
<td>REORG (Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional. You must also include a DFSURWF1 DD statement that is not DD DUMMY.</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

**RECONx DD**

The data sets named in the RECONx DD statements define the DBRC RECON data sets. The RECONx DD statements are required for HALDBs. The RECONx DD statements are optional for full-function databases. BMC strongly recommends that you do not include them in the JCL for full-function databases. Instead, add an MDALIB that contains the RECON data sets to your STEPLIB concatenation. RECON MDAs can be stored in the RESLIB.

Table 87 describes the RECONx DD requirements by command.
SEQERROR DD

The data set named in the SEQERROR DD statement defines the sequential data set to which the Reload function writes duplicate or out-of-sequence segments if you specified SEQERROR(SAVE) and either of these errors are encountered in the database.

Table 88 describes the SEQERROR DD requirements by command.

Table 88  SEQERROR DD requirements by command

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Optional. If you do not specify the SEQERROR DD statement, the Reload function dynamically allocates this data set from the work data set allocation parameters specified in the PDX. See “SEQERROR” on page 537. If you do not provide this DD statement and you do not use a PDX or use a global options module, the Reload function job continues, does not allocate the data set, and ends with a return code of 16. Note: The SEQERROR DD statement is not valid if you specify PARALLEL(YES) or accept it as the default.</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>n/a</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

SORTIN DD

Defines the work data sets created during database load, reload, or scan. If the IBM Work File 1 Generator was used when this data set was created, you must code the DCB information on the DD statement.

- If you specified SORTIN(N), this data set must contain only BMC-format logical relationship records. It must not contain secondary index records.

- If you specified SORTIN(Y), this data set must contain IMS-format logical relationship records; it can also contain secondary index records.

Table 89 on page 378 describes the SORTIN DD requirements by command.
SORTWK DD

The data set named in the SORTWK DD statement describes the sort work data sets used by the sort product. BMC recommends that you use the SORTWORK keyword (for all functions except the Prefix Resolution/Update function) to dynamically allocate SORTWKnn DD statements. For more information, see “SORTWORK” on page 545.

For more information about the SORTWKnn DD statement, see Chapter 8, “Index Build function.”

Table 90 describes the SORTWK DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>LOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Required if you do not supply a DFSURWF1 DD statement.</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>

STEPLIB DD

The data sets named in the STEPLIB DD statement describe the libraries containing the BMC product and IMS load modules. This statement can also describe IMS user exit routines or user application programs. If your site’s sort program library is not specified in the LINKLIST, you must specify it here. Always concatenate the BMC product library ahead of the IMS library.
**NOTE**

If you are using the Online Reorg function, the load library must be APF authorized. If you are using MAXM Reorg/Online or MAXM Reorg/EP solutions, all libraries in the STEPLIB concatenation must be APF authorized or you must have started the Database Utilities Subsystem (DBUSS). For more information, see the *BMC Products for IMS Installation Guide*.

Table 91 describes the STEPLIB DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function and Reorg function)</td>
<td>Required. For the Online Reorg function, all libraries in the STEPLIB must be APF-authorized.</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>Required</td>
</tr>
<tr>
<td>RELOAD</td>
<td>Required</td>
</tr>
<tr>
<td>BUILD</td>
<td>Required</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Required</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a; the requirement is for the REORG command.</td>
</tr>
</tbody>
</table>

**SYSOUT DD**

The data set named in the SYSOUT DD statement defines the sort message data set for logical relationship processing.

Table 92 describes the SYSOUT DD requirements by command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function and Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Required</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>n/a</td>
</tr>
</tbody>
</table>
The data set named in the SYSUDUMP DD statement defines a system dump data set. Normally it defines a SYSOUT= data set.

Table 93 describes the SYSUDUMP DD requirements by command.

**NOTE**
Abbreviated dumps and Abend-Aid dumps do not provide the information that BMC needs for problem resolution. Always take a complete dump.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>REORG (Online Reorg function and Reorg function)</td>
<td>n/a</td>
</tr>
<tr>
<td>UNLOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>RELOAD</td>
<td>n/a</td>
</tr>
<tr>
<td>BUILD</td>
<td>n/a</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>Optional. If the Prefix Resolution/Update function encounters any problems, it prints an error message to the PLUSLIST data set and causes an abend. The dump might be necessary to resolve the problem.</td>
</tr>
<tr>
<td>Image Copy function</td>
<td>Optional. The SYSUDUMP DD statement does not degrade performance or add storage requirements when the Image Copy function completes successfully. The Image Copy function dynamically allocates a SYSUDUMP DD statement if you do not have one in your JCL. Having this data set aids BMC in faster problem resolution. If you do not want this data set automatically allocated, code a SYSUDUMP DD DUMMY statement in your JCL.</td>
</tr>
</tbody>
</table>
User-controlled options

Options let you tune and control processing. This chapter describes the options and the tasks they accomplish. This chapter includes the following topics:

Overview ................................................................. 384
How the utilities build options .................................. 385
Setting options ......................................................... 387
  Online Reorg function ........................................ 388
  Reorg function .................................................... 388
  Individual MAXM Reorg functions ......................... 388
Commands and keywords overview ............................. 388
  PLUSIN control statement syntax .......................... 404
Keyword descriptions .............................................. 406
  ABNDNO .................................................................. 406
  AGN .................................................................... 407
  ALLOWLC ............................................................ 407
  ALLOWLP ............................................................. 407
  AMSOUT .............................................................. 408
  AOI ..................................................................... 409
  AOIUSER .............................................................. 410
  AUT thERR ............................................................ 410
  AUTOSTOP ............................................................ 411
  BIPF ................................................................... 413
  BUFSpace ............................................................. 413
  CKUPDATE ........................................................... 414
  COMPRESS, EXPAND ............................................ 415
  CREORG ............................................................... 419
  DBD ................................................................... 420
  DBDSgrp .............................................................. 420
  DBR .................................................................... 421
  DBRC (Index Build and Reorg functions) ................. 422
  DBRC (Prefix Resolution/Update function) ............... 423
  DDN ................................................................... 423
  DESC (Image Copy function) ................................. 424
  DFSDBUX1 .......................................................... 424
  DFSURGU1 .......................................................... 425
  DYNALLOC ........................................................... 430
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>EOJACCES</td>
<td>433</td>
</tr>
<tr>
<td>FBFF</td>
<td>434</td>
</tr>
<tr>
<td>FFTOHAL</td>
<td>434</td>
</tr>
<tr>
<td>FILSZ</td>
<td>437</td>
</tr>
<tr>
<td>FIXED</td>
<td>437</td>
</tr>
<tr>
<td>FMTALL</td>
<td>438</td>
</tr>
<tr>
<td>FMTRAAD</td>
<td>439</td>
</tr>
<tr>
<td>FORCELP</td>
<td>439</td>
</tr>
<tr>
<td>FREQ</td>
<td>440</td>
</tr>
<tr>
<td>FSPF</td>
<td>441</td>
</tr>
<tr>
<td>FUNC</td>
<td>442</td>
</tr>
<tr>
<td>GSGNAME</td>
<td>442</td>
</tr>
<tr>
<td>HDAMFSPF</td>
<td>443</td>
</tr>
<tr>
<td>HDSORT</td>
<td>443</td>
</tr>
<tr>
<td>HIPO</td>
<td>446</td>
</tr>
<tr>
<td>HIST</td>
<td>447</td>
</tr>
<tr>
<td>HSAMOUT</td>
<td>448</td>
</tr>
<tr>
<td>IC</td>
<td>449</td>
</tr>
<tr>
<td>ICACC</td>
<td>449</td>
</tr>
<tr>
<td>ICALLOC</td>
<td>450</td>
</tr>
<tr>
<td>ICAVGREC</td>
<td>451</td>
</tr>
<tr>
<td>ICBUF</td>
<td>451</td>
</tr>
<tr>
<td>ICCATF</td>
<td>452</td>
</tr>
<tr>
<td>ICDATAACL</td>
<td>453</td>
</tr>
<tr>
<td>ICDSCL</td>
<td>454</td>
</tr>
<tr>
<td>ICEXPDT</td>
<td>454</td>
</tr>
<tr>
<td>ICGDG</td>
<td>455</td>
</tr>
<tr>
<td>ICLIKE</td>
<td>455</td>
</tr>
<tr>
<td>ICMGMTCL</td>
<td>456</td>
</tr>
<tr>
<td>ICNEEDED</td>
<td>457</td>
</tr>
<tr>
<td>ICP</td>
<td>458</td>
</tr>
<tr>
<td>ICPREF</td>
<td>459</td>
</tr>
<tr>
<td>ICETPD</td>
<td>460</td>
</tr>
<tr>
<td>ICSPACE</td>
<td>461</td>
</tr>
<tr>
<td>ICSTORCL</td>
<td>462</td>
</tr>
<tr>
<td>ICUNIT</td>
<td>463</td>
</tr>
<tr>
<td>ICVOLCNT</td>
<td>463</td>
</tr>
<tr>
<td>ICVOLSER</td>
<td>464</td>
</tr>
<tr>
<td>IDCAMS</td>
<td>464</td>
</tr>
<tr>
<td>ILDS</td>
<td>464</td>
</tr>
<tr>
<td>IMSID</td>
<td>468</td>
</tr>
<tr>
<td>LIM</td>
<td>470</td>
</tr>
<tr>
<td>LIMITS</td>
<td>471</td>
</tr>
<tr>
<td>IMSID</td>
<td>471</td>
</tr>
<tr>
<td>INDD</td>
<td>472</td>
</tr>
<tr>
<td>INDDX</td>
<td>472</td>
</tr>
<tr>
<td>INPUT</td>
<td>473</td>
</tr>
<tr>
<td>INST</td>
<td>476</td>
</tr>
<tr>
<td>IRLMGPR</td>
<td>477</td>
</tr>
<tr>
<td>KEYS</td>
<td>478</td>
</tr>
<tr>
<td>LIM</td>
<td>478</td>
</tr>
<tr>
<td>LIMITS</td>
<td>479</td>
</tr>
<tr>
<td>Option</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>LOGDSN</td>
<td>483</td>
</tr>
<tr>
<td>LOGEXPDT, LOGRETPD</td>
<td>485</td>
</tr>
<tr>
<td>LOGSMS</td>
<td>485</td>
</tr>
<tr>
<td>LOGUNIT</td>
<td>486</td>
</tr>
<tr>
<td>LOGSPACE</td>
<td>486</td>
</tr>
<tr>
<td>LPCK</td>
<td>487</td>
</tr>
<tr>
<td>MBI</td>
<td>488</td>
</tr>
<tr>
<td>MINREC</td>
<td>489</td>
</tr>
<tr>
<td>MODE</td>
<td>489</td>
</tr>
<tr>
<td>MONITOR, MONUSERS, ROUTCODE, DESCCODE</td>
<td>490</td>
</tr>
<tr>
<td>NAME</td>
<td>493</td>
</tr>
<tr>
<td>NEWDBD</td>
<td>494</td>
</tr>
<tr>
<td>NOAUTH</td>
<td>494</td>
</tr>
<tr>
<td>NOTIFY</td>
<td>495</td>
</tr>
<tr>
<td>OLDDBD</td>
<td>496</td>
</tr>
<tr>
<td>OSAMINIT</td>
<td>499</td>
</tr>
<tr>
<td>OSAMMAX</td>
<td>500</td>
</tr>
<tr>
<td>OUTDD</td>
<td>501</td>
</tr>
<tr>
<td>OUTDDX</td>
<td>501</td>
</tr>
<tr>
<td>OVFLOONLY</td>
<td>503</td>
</tr>
<tr>
<td>PADCHAR</td>
<td>503</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>504</td>
</tr>
<tr>
<td>PARMBLK</td>
<td>507</td>
</tr>
<tr>
<td>PART (database partitioning)</td>
<td>508</td>
</tr>
<tr>
<td>PCP</td>
<td>517</td>
</tr>
<tr>
<td>PDX (Unload, Reload, and Index Build functions)</td>
<td>518</td>
</tr>
<tr>
<td>PDX (Image Copy function)</td>
<td>518</td>
</tr>
<tr>
<td>PDX (Prefix Resolution/Update function)</td>
<td>519</td>
</tr>
<tr>
<td>PDX (Online/Defrag function)</td>
<td>519</td>
</tr>
<tr>
<td>PDXPARMS</td>
<td>520</td>
</tr>
<tr>
<td>PRIMIND</td>
<td>520</td>
</tr>
<tr>
<td>PRTOLC</td>
<td>521</td>
</tr>
<tr>
<td>PRTOLP</td>
<td>521</td>
</tr>
<tr>
<td>xxxPRINT</td>
<td>522</td>
</tr>
<tr>
<td>PSB</td>
<td>523</td>
</tr>
<tr>
<td>PTRCHECK</td>
<td>524</td>
</tr>
<tr>
<td>PTRERROR</td>
<td>524</td>
</tr>
<tr>
<td>REORGMMSG</td>
<td>525</td>
</tr>
<tr>
<td>REPORTS</td>
<td>526</td>
</tr>
<tr>
<td>RESUME</td>
<td>528</td>
</tr>
<tr>
<td>RMBYTES</td>
<td>529</td>
</tr>
<tr>
<td>ROUTCDE (Image Copy function)</td>
<td>530</td>
</tr>
<tr>
<td>SCAN—space search method for HIDAM and PHIDAM</td>
<td>531</td>
</tr>
<tr>
<td>SCAN—space search method for HDAM and PHDAM</td>
<td>531</td>
</tr>
<tr>
<td>SECINDEX</td>
<td>534</td>
</tr>
<tr>
<td>SEGPOS</td>
<td>535</td>
</tr>
<tr>
<td>SEGPOSO</td>
<td>536</td>
</tr>
<tr>
<td>SEGSEL</td>
<td>536</td>
</tr>
<tr>
<td>SEQERROR</td>
<td>537</td>
</tr>
</tbody>
</table>
When you specify the SHARE(YES,UPDATE) keyword on the REORG command, the Online Reorg function calls the Reorg function to manage the reorganization. When you issue the REORG command and do not specify SHARE(YES,UPDATE), the Reorg function manages the reorganization by using the Unload function, the Reload function, the Index Build function, and the prefix resolution and update function.

When the Online Reorg function or the Reorg function control a reorganization, some optional keywords in the underlying functions must be assigned particular values. The Reorg function sets these values at execution time by passing them directly to the utilities. These values cannot be overridden. For more information, see “MAXM Reorg keywords and the REORG command” on page 575.
You can use an individual utility to accomplish a specific reorganization task. For example, you can build indexes by executing an Index Build function job.

## How the utilities build options

This section describes how the utilities build options. For information about how the Image Copy function builds options, see “Global options modules” on page 280.

At execution time, each utility follows an iterative process to determine which options values you want to use for a particular database. Each succeeding step replaces keyword values that were determined in the previous step:

1. Each utility builds an initial parameter table from the internally coded default values.

2. Each utility searches for customization modules, as shown in Table 94.

### Table 94 Customization modules

<table>
<thead>
<tr>
<th>Product</th>
<th>First searches for</th>
<th>Then searches for</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online Reorg function (with limited prefix resolution and update support for logical relationships; for limitations, see page 72)</td>
<td>FRF$GLBL</td>
<td>n/a</td>
</tr>
<tr>
<td>Reorg function</td>
<td>FRF$GLBL</td>
<td>n/a</td>
</tr>
<tr>
<td>Unload function</td>
<td>ULPPARMS</td>
<td>ULP@imsi</td>
</tr>
<tr>
<td>Reload function</td>
<td>LDPPARMS</td>
<td>LDP@imsi</td>
</tr>
<tr>
<td>Index Build function</td>
<td>SIUPARMS</td>
<td>SIU@imsi</td>
</tr>
<tr>
<td>Prefix Resolution/Update function</td>
<td>PRPPARMS</td>
<td>PRP@imsi</td>
</tr>
</tbody>
</table>

Each utility merges the values coded in the modules it finds first to replace values in the parameter table.

3. The Unload function and the Reload function search the PDX for a PDX customization member that is associated with this DBD and IMSID. If the functions cannot find a PDX customization member, they search the PDX named in the JCL or the global options table for a PDX customization member named DFLT. If the DFLT member exists, it was created with an earlier release of a BMC utility.

4. Each utility uses the DBD name, DBRC, and IMSID values coded on the PARM parameter in the EXEC statement to add to or replace the values in the parameter table.
5. Each utility uses any values coded in the PLUSIN control statement data set to add to or replace the keywords built in the previous steps.

The result of these steps is the options table that the utility uses during database processing. The global options are not modified.

With the Online Reorg function and the Reorg function, you can combine the global options module and commands for specifying keywords for maximum efficiency.

With the Unload function and the Reload function, you can combine the global options module, the PDX options member, and command methods for specifying options for maximum efficiency. You can use the xxxPARMS options module to specify the values you want to use for most of the databases at your site. For particular keywords you want to use only on a single IMS system (for example, a test system), you can use the xxx@imsi options module. For databases that need different values from those you specified in the xxxPARMS or xxx@imsi modules, you can create a PDX options member for those DBDs. For jobs that require different values from those you usually use for the database, you can specify the values in the PLUSIN data set.

With the Index Build function, you can combine the global options module and command methods for specifying options for maximum efficiency. You can use the SIUPARMS options module to specify the values you want to use for most of the databases at your site. For particular keywords you want to use only on a single IMS system (for example, a test system), you can use the SIU@imsi options module. For jobs that require different values from those you usually use for the database, you can specify the values in the PLUSIN data set.

**NOTE**

The environment setup member (DLIGSET0) contains values for MAXM Reorg solution parameters that are not available in xxxPARMS members. The values that you specify in DLIGSET0 are used at execution time. If you use a DBUSS, you must specify the DBUSS name in DLIGSET0. The DBUSSID keyword is not available in the PLUSIN control statement data set, and if you specified a DBUSS name in an xxxPARMS member, it will be ignored.

The Options in Effect report lists the user-controlled options in effect for the current run.

Figure 84 on page 387 shows how the options are built.
Setting options

You can set options at several levels, depending on which utility you execute. All functions have internal defaults. You can use these defaults without creating global options modules. You can set job-level options by including the keywords in the PLUSIN control statement data set. The options that you specify in PLUSIN are in effect only for that job.

You might want to set certain option values to meet your site’s specific needs. This section explains how to create customized options modules.

NOTE
For information about how to set options for the Image Copy function, see “Global options modules” on page 280.
Online Reorg function

Create the environment setup member DLIGSET0 by executing the sample member #DLIGLBL. This member sets installation-wide options for the Online Reorg function. For more information about the environment setup member, see the BMC Products for IMS Installation Guide.

For more information about the global options module that is available for the Online Reorg function, see “Global options module” on page 124. No ISPF interface is available for the Online Reorg function.

Reorg function

Create the FRF$GLBL global options module by executing member #FRFGLBL in the sample library. For more information, see “Global options module” on page 101. No ISPF interface is available for the Reorg function.

Individual MAXM Reorg functions

You can build global options modules for the Unload function, the Reload function, the Index Build function, and the Prefix Resolution/Update function by executing batch jobs or by using the optional ISPF interfaces available with the Database Utilities.

For more information, see the following sections:

- Unload function—“Global options modules” on page 181
- Reload function—“Global options modules” on page 219
- Index Build function—“Global options modules” on page 251
- Prefix Resolution/Update function—“Global options modules” on page 280

Commands and keywords overview

Each function supports commands and keywords to control processing. Table 95 on page 389 shows the functions and valid commands.
### Table 95 MAXM Reorg/Online commands

<table>
<thead>
<tr>
<th>Function</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online Reorg function</td>
<td>REORG</td>
<td>Reorganize a database while updates are being made by an online IMS or CICS/DBCTL system by using the Reorg function, the Unload function, the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reload function, the Index Build function, Prefix Resolution/Update function, and Image Copy function.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The Online/Reorg function does not support HALDBs with logical relationships.</td>
</tr>
<tr>
<td>COPY</td>
<td></td>
<td>Make a copy of the database during the first phase of the reorganization. For more information, see Chapter 9, “Copy function.”</td>
</tr>
<tr>
<td>Reorg function</td>
<td>REORG</td>
<td>Reorganize a database by using Unload function, Reload function, Index Build function, and the prefix resolution and update function.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The Reorg function does not support HALDBs with logical relationships.</td>
</tr>
<tr>
<td>Unload function</td>
<td>UNLOAD</td>
<td>Unload the records in a database.</td>
</tr>
<tr>
<td></td>
<td>DBSELECT</td>
<td>Retrieve selected records from a database. For more information, see “Database Selection facility” on page 192.</td>
</tr>
<tr>
<td></td>
<td>TRACE</td>
<td>Use the API. For more information, see “API” on page 199.</td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>USERREAD</td>
<td>Use the API to load a database initially. For more information, see “API” on page 234.</td>
</tr>
<tr>
<td>Reload function</td>
<td>RELOAD</td>
<td>Reload the records in a database.</td>
</tr>
<tr>
<td></td>
<td>SORT</td>
<td>Sort unloaded records before reloading them by using the stand-alone HD Sort utility. For more information, see “HD Sort utility” on page 227.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To sort and reload a database, specify the HDSORT keyword with the RELOAD command.</td>
</tr>
<tr>
<td></td>
<td>USERLOAD</td>
<td>Use the API to load a database initially. For more information, see “API” on page 234.</td>
</tr>
<tr>
<td>Index Build function</td>
<td>BUILD</td>
<td>Build indexes (primary and/or secondary) for a database.</td>
</tr>
<tr>
<td></td>
<td>BUILDALL</td>
<td></td>
</tr>
<tr>
<td>Image Copy function</td>
<td>IC</td>
<td>Create output image copies during the database reorganization process. Keywords are specified in the ICPSYSIN data set.</td>
</tr>
<tr>
<td></td>
<td>AIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GLBL</td>
<td>Set global and group options for the Image Copy function. Keywords are specified in the ICPSYSIN data set.</td>
</tr>
<tr>
<td></td>
<td>GROUP</td>
<td></td>
</tr>
<tr>
<td>Online/Defrag function</td>
<td>ANALYZE</td>
<td>Perform an Online/Defrag Record Analysis.</td>
</tr>
<tr>
<td></td>
<td>OREORG</td>
<td>Perform an Online/Defrag reorganization.</td>
</tr>
</tbody>
</table>

Table 96 on page 390 lists (in alphabetical order) the keywords you can set, accepted and default values, and the pages where descriptions appear.

---

**NOTE**

You can enter abbreviated values for some keywords. The abbreviated values are underlined. Unless noted otherwise, keywords valid for the REORG command are applicable for the Online Reorg function and the Reorg function.
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABNDNO</td>
<td>Prefix Resolution/Update function</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>406</td>
<td>Cause the Prefix Resolution/Update function to abend if no output records are written to any output file. Specify N to permit the condition to occur.</td>
</tr>
<tr>
<td>AGN</td>
<td>OREORG (Online Reorg function)</td>
<td>(agnName)</td>
<td>none</td>
<td>407</td>
<td>Specify an application group name for the BMP in the Online Reorg function.</td>
</tr>
<tr>
<td>ALLOWLC</td>
<td>Prefix Resolution/Update function</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>407</td>
<td>Allow logical children with no logical parents. Abend if condition occurs with default.</td>
</tr>
<tr>
<td>ALLOWLP</td>
<td>Prefix Resolution/Update function</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>407</td>
<td>Allow logical parents with no logical children. Do not abend if condition occurs with default.</td>
</tr>
<tr>
<td>AOI</td>
<td>REORG (Reorg function)</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>409</td>
<td>Use the Online Reorg function automated operator interface during a Reorg function reorganization.</td>
</tr>
<tr>
<td>AOIUSER</td>
<td>COPY REORG (Online Reorg function)</td>
<td>(progName)</td>
<td>none</td>
<td>410</td>
<td>Invoke a user exit from within the AOI.</td>
</tr>
<tr>
<td>AUTHERR</td>
<td>REORG (Reorg function)</td>
<td>(WTOR) (ABEND)</td>
<td>(ABEND)</td>
<td>410</td>
<td>Specify the Reorg function action if DBRC authorization fails.</td>
</tr>
<tr>
<td>BUFSPACE</td>
<td>REORG UNLOAD</td>
<td>(1-999)</td>
<td>(20)</td>
<td>413</td>
<td>Specify maximum amount of buffer space (in MB) for the unload step.</td>
</tr>
</tbody>
</table>
### Table 96  Keywords (part 2 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKUPDATE</td>
<td>RELOAD</td>
<td>(YES,STOPnmmm)</td>
<td>(YES,STOP4094)</td>
<td>414</td>
<td>Specify whether to check the RECON data sets for updates after the unload file was created.</td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td>(YES,OKAYnmmm)</td>
<td>(YES,OKAYnmmm)</td>
<td>420</td>
<td>Specify conditional reorganization option for the DBA Toolkit.</td>
</tr>
<tr>
<td></td>
<td>(Reorg function)</td>
<td>(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td>(YES,STOPnmmm)</td>
<td>(YES,STOPnmmm)</td>
<td>415</td>
<td>Compress segments during reload.</td>
</tr>
<tr>
<td></td>
<td>(YES,OKAYnmmm)</td>
<td>(YES,OKAYnmmm)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPRESS</td>
<td>RELOAD</td>
<td>(YES)</td>
<td>(YES)</td>
<td>415</td>
<td>Compress segments during reload.</td>
</tr>
<tr>
<td>CREORG</td>
<td>UNLOAD</td>
<td>(YES,OKAYnmmm)</td>
<td>(YES,OKAYnmmm)</td>
<td>420</td>
<td>Specify conditional reorganization option for the DBA Toolkit.</td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td>(YES,STOPnmmm)</td>
<td>(YES,STOPnmmm)</td>
<td>415</td>
<td>Compress segments during reload.</td>
</tr>
<tr>
<td></td>
<td>(YES,OKAYnmmm)</td>
<td>(YES,OKAYnmmm)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(NO)</td>
<td>(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBD</td>
<td>BUILD</td>
<td>(dbdName1, dbdName2,...)</td>
<td>none</td>
<td>420</td>
<td>Specify the DBD name for the primary DBD.</td>
</tr>
<tr>
<td></td>
<td>COPY</td>
<td>(dbdName)</td>
<td>none</td>
<td>420</td>
<td>Specify the DBD name for the primary DBD.</td>
</tr>
<tr>
<td></td>
<td>IC AIC (ICPSYSIN)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBDSGRP</td>
<td>GROUP</td>
<td>(dbdsName)</td>
<td>none</td>
<td>420</td>
<td>Generate image copy data sets for all data set groups defined in the database data set group.</td>
</tr>
<tr>
<td></td>
<td>(ICPSYSIN)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBR</td>
<td>OREORG</td>
<td>(FEOV)</td>
<td>(FEOV)</td>
<td>421</td>
<td>Specify whether to switch the OLDS for the /DBR at the end of the reorganization.</td>
</tr>
<tr>
<td></td>
<td>(Online Reorg function)</td>
<td>(NOFEOV)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBRC</td>
<td>BUILD</td>
<td>(YES)</td>
<td>Use the DBRC specification in the IMSCTRL macro.</td>
<td>422</td>
<td>Specify whether to invoke DBRC. For HALDB, DBRC(YES) is required.</td>
</tr>
<tr>
<td></td>
<td>COPY</td>
<td>(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>COPY</td>
<td>(Y)</td>
<td>none</td>
<td>423</td>
<td>Invoke DBRC when there is no region controller. Specify N to invoke only by region controller.</td>
</tr>
<tr>
<td></td>
<td>Prefix Resolution/Update function</td>
<td>(N)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBRC</td>
<td>COPY</td>
<td>(Y)</td>
<td>none</td>
<td>423</td>
<td>Invoke DBRC when there is no region controller. Specify N to invoke only by region controller.</td>
</tr>
<tr>
<td></td>
<td>Prefix Resolution/Update function</td>
<td>(N)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DDN</td>
<td>IC</td>
<td>(ddname)</td>
<td>none</td>
<td>423</td>
<td>Specify the ddname associated with the data set group to be processed.</td>
</tr>
<tr>
<td></td>
<td>AIC (ICPSYSIN)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DESC</td>
<td>Prefix Resolution/Update function</td>
<td>(nn,...)</td>
<td>(7)</td>
<td>490</td>
<td>Consult MVS Supervisor Macros Manual under WTO and WTOR macros for proper designation.</td>
</tr>
<tr>
<td></td>
<td>DESC</td>
<td>(nn,...)</td>
<td>(7)</td>
<td>424</td>
<td>Specify the descriptor code numbers that designate the manner in which MVS is to manage the highlighting and retention of monitoring messages issued to MVS consoles.</td>
</tr>
<tr>
<td></td>
<td>GLBL (ICPSYSIN)</td>
<td>nn = 1–11 max 10 entries</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 96  Keywords (part 3 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESC CODE</td>
<td>BUILD DBSCAN RELOAD REORG UNLOAD</td>
<td>(nn,...) max 10 entries</td>
<td>(7)</td>
<td>490</td>
<td>Specify descriptor codes used to route messages to the console.</td>
</tr>
<tr>
<td>DFSDBUX1</td>
<td>RELOAD UNLOAD</td>
<td>(YES) (NO)</td>
<td>(YES)</td>
<td>424</td>
<td>Specify whether to invoke the DFSDBUX1 user exit.</td>
</tr>
<tr>
<td>DFSDBUX1</td>
<td>REORG (Reorg function)</td>
<td>(YES,YES) (NO,NO)</td>
<td>(YES,YES)</td>
<td>424</td>
<td>Specify whether to invoke the DFSDBUX1 user exit.</td>
</tr>
<tr>
<td>DFSURGU1</td>
<td>UNLOAD</td>
<td>(LONG) (SHORT) (XSHORT) (FRMT1) (FRMT2) (FRMT3)</td>
<td>(LONG)</td>
<td>425</td>
<td>Select the format of HD Unload records.</td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>BUILD DBSCAN RELOAD UNLOAD</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>430</td>
<td>Dynamically allocate database data sets using the data set names specified in the RECON data sets.</td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>COPY REORG</td>
<td>(YES) (NO) (YES,YES, x) (YES,NO, x) (NO,YES)</td>
<td>(NO)</td>
<td>430</td>
<td>Dynamically allocate all database data sets using the data set names specified in the RECON data sets, and give output data sets unique names.</td>
</tr>
<tr>
<td>EOJACCES</td>
<td>REORG</td>
<td>(EX) (UP) (RD) (RO)</td>
<td></td>
<td>433</td>
<td>Access level for final /STA command.</td>
</tr>
<tr>
<td>EXPAND</td>
<td>REORG UNLOAD</td>
<td>(YES) (NO)</td>
<td>REORG:(NO) UNLOAD:(YES)</td>
<td>415</td>
<td>Expand compressed segments.</td>
</tr>
<tr>
<td>FBFF</td>
<td>RELOAD USERLOAD REORG</td>
<td>(n1,n2,...n10)</td>
<td></td>
<td>434</td>
<td>Specify an override value for the free block frequency factor specified in the DBD.</td>
</tr>
<tr>
<td>FFTOHAL</td>
<td>LOAD UNLOAD</td>
<td>(NO) (YES)</td>
<td>NO</td>
<td>434</td>
<td>Creates an Unload file usable by BMC to migrate Full Function logically related databases to HALDB.</td>
</tr>
<tr>
<td>FILSZ</td>
<td>Index Build function</td>
<td>(INDEX) (NO)</td>
<td>INDEX</td>
<td>437</td>
<td>Specify an estimated number of records for each index file.</td>
</tr>
</tbody>
</table>
### Table 96  Keywords (part 4 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIXED</td>
<td>Prefix</td>
<td>Resolution/Update function</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>434</td>
</tr>
<tr>
<td>FMTALL</td>
<td>RELOAD REORG GROUP USERLOAD</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>438</td>
<td>Format all data sets.</td>
</tr>
<tr>
<td>FMTRAA</td>
<td>RELOAD REORG</td>
<td>(YES) (NO)</td>
<td>(YES)</td>
<td>439</td>
<td>Format the HDAM or PHDAM root addressable area.</td>
</tr>
<tr>
<td>FORCEL P</td>
<td>Prefix</td>
<td>Resolution/Update function</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>439</td>
</tr>
<tr>
<td>FREQ</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>(nnnnnnnn) nnnnnnnn = 0–99999999</td>
<td>(10000)</td>
<td>440</td>
<td>Specify how often the Image Copy function writes a message to the processing log or system console about the number of records it has read or written.</td>
</tr>
<tr>
<td>FRFPRINT</td>
<td>REORG</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>522</td>
<td>Specify the data set that will contain the Online Reorg function and the Reorg function reports and processing output.</td>
</tr>
<tr>
<td>FSPF</td>
<td>RELOAD USERLOAD REORG OREOR G ANALY</td>
<td>(n1,n2...n10)</td>
<td>95</td>
<td>441</td>
<td>Specify a percentage of free space to qualify a block as free.</td>
</tr>
<tr>
<td>FUNC</td>
<td>GROUP</td>
<td>(IC) (AIC)</td>
<td>none</td>
<td>442</td>
<td>Identify the Image Copy function to perform for all data set groups defined with the DBDSGRP keyword.</td>
</tr>
<tr>
<td>GSGNAME</td>
<td>COPY BUILD Prefix Resolution/Update function</td>
<td>(gsgName)</td>
<td>Use the GSG name used in system generation.</td>
<td>442</td>
<td>Use a global service group (GSG) name for remote site recovery (RSR).</td>
</tr>
<tr>
<td>Keyword</td>
<td>Command</td>
<td>Accepted values</td>
<td>Default values</td>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
<td>-----------------</td>
<td>----------------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>HDAMFSPF</td>
<td>RELOAD</td>
<td>(IGNORE)</td>
<td>(NO)</td>
<td>443</td>
<td>Ignore the free space percentage factor (FSPF).</td>
</tr>
<tr>
<td>HDSORT</td>
<td>RELOAD</td>
<td>(YES)</td>
<td>(NO)</td>
<td>443</td>
<td>Invoke the HD Sort utility or the BMC RAP Sequencer.</td>
</tr>
<tr>
<td>HIPO</td>
<td></td>
<td>(Y)</td>
<td>(N)</td>
<td>446</td>
<td>Pass output from Parent-Child Resolution directly to Logical Twin Resolution using virtual storage internal buffering.</td>
</tr>
<tr>
<td>HIST</td>
<td>GLBL</td>
<td>(nnnn)</td>
<td>(9999)</td>
<td>447</td>
<td>Specify a limit for the generations of historical information that the Image Copy function maintains in the PDX for each execution of the Image Copy function.</td>
</tr>
<tr>
<td>HSAMOUT</td>
<td>UNLOAD</td>
<td>(dbname)</td>
<td>blank</td>
<td>448</td>
<td>Write an HSAM database.</td>
</tr>
<tr>
<td>IC</td>
<td>GROUP</td>
<td>(ddname, ddname,...)</td>
<td>none</td>
<td>449</td>
<td>Specify the ddname of each output image copy data set to write.</td>
</tr>
<tr>
<td>ICACC</td>
<td>GLBL</td>
<td>(1)</td>
<td>(1)</td>
<td>449</td>
<td>Specify the access method (1=SAM, 2=EXCP) to use for writing output image copy data sets.</td>
</tr>
<tr>
<td>ICAVGREC</td>
<td>GLBL</td>
<td>()</td>
<td>()</td>
<td>451</td>
<td>Specify the allocation units (blocks, logical records, or multiples of logical records) for DASD space when dynamically allocating new SMS output image copy data sets.</td>
</tr>
<tr>
<td>ICBUF</td>
<td>GLBL</td>
<td>(0)</td>
<td>(4)</td>
<td>451</td>
<td>Specify the output image copy buffer option.</td>
</tr>
<tr>
<td>Keyword</td>
<td>Command</td>
<td>Accepted values</td>
<td>Default values</td>
<td>Page</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>--------------------</td>
<td>-------------------</td>
<td>----------------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ICCATF</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>452</td>
<td>Create catalog entries for dynamically cataloged output image copies.</td>
</tr>
<tr>
<td>ICDATACL</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>() (dataclas)</td>
<td>()</td>
<td>453</td>
<td>Specify the DATACLAS (data class) when you dynamically allocate new SMS image copy output data sets.</td>
</tr>
<tr>
<td>ICDSCB</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>() (dscbName)</td>
<td>none</td>
<td>454</td>
<td>Specify the name of the data set control block (DSCB) to use as a model for the generation data group (GDG) when you dynamically allocate the output image copy data sets.</td>
</tr>
<tr>
<td>ICEXPDT</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>(yyddd) (yyyyddd)</td>
<td>none</td>
<td>454</td>
<td>Specify an expiration date for output image copy data sets when they are dynamically allocated.</td>
</tr>
<tr>
<td>ICGDG</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>(N) (Y)</td>
<td>(N)</td>
<td>455</td>
<td>Create image copy data sets as GDGs when the output image copy data sets are dynamically allocated.</td>
</tr>
<tr>
<td>ICLIKE</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>() (dataSetName)</td>
<td>()</td>
<td>455</td>
<td>Specify the data set name of an existing SMS data set to use as a model when you dynamically allocate a new SMS image copy output data set.</td>
</tr>
<tr>
<td>ICMGMTCL</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>() (className)</td>
<td>()</td>
<td>456</td>
<td>Specify the name of the MGMTCLAS (management class) to use when you dynamically allocate new SMS image copy output data sets.</td>
</tr>
<tr>
<td>ICNEEDED</td>
<td>BUILD REORG</td>
<td>(ON) (OFF)</td>
<td>(ON) for RECOV databases (OFF) for NONRECOV databases</td>
<td>457</td>
<td>Specify whether to turn off the ICNEEDED flag for all NONRECOV index DBDs (BUILD) or all NONRECOV DBDs (REORG).</td>
</tr>
</tbody>
</table>
### Table 96  Keywords (part 7 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICP</td>
<td>REORG</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>458</td>
<td>Invoke the Image Copy function.</td>
</tr>
<tr>
<td>ICPREF</td>
<td>GLBL GROUP IC AIC</td>
<td>(dsnPrefix)</td>
<td>none</td>
<td>459</td>
<td>Identify a data set name prefix for the output image copy data set.</td>
</tr>
<tr>
<td>ICRETPD</td>
<td>GLBL GROUP IC AIC</td>
<td>(value) value = 0–9999</td>
<td>none</td>
<td>460</td>
<td>Specify a retention period (in days) for output image copy data sets that are dynamically allocated.</td>
</tr>
<tr>
<td>ICSPACE</td>
<td>GLBL GROUP IC AIC</td>
<td>() (unit,pri,sec) (unit,pri)</td>
<td>()</td>
<td>461</td>
<td>Specify the amount of DASD space to allocate for a new non-SMS data set or to override the space allocation defined in the DATACLAS (data class) for an SMS data set.</td>
</tr>
<tr>
<td>ICSTORCL</td>
<td>GLBL GROUP IC AIC</td>
<td>() (className)</td>
<td>()</td>
<td>462</td>
<td>Specify the name of a STORCLAS (storage class) to use when you dynamically allocate new SMS image copy output data sets.</td>
</tr>
<tr>
<td>ICUNIT</td>
<td>GLBL GROUP IC AIC</td>
<td>(unitName) (unitName,count)</td>
<td>(TAPE,1)</td>
<td>463</td>
<td>Specify the device unit name to use when dynamically allocating each output image copy data sets. Also specify the number of physical units to allocate.</td>
</tr>
<tr>
<td>ICVOLCNT</td>
<td>GLBL GROUP IC AIC</td>
<td>(value) value = 0–255</td>
<td>(0)</td>
<td>463</td>
<td>Override your installation’s MVS specification on the maximum number of volumes allowed for an output image copy data set when they are dynamically allocated.</td>
</tr>
<tr>
<td>ICVOLSER</td>
<td>GLBL GROUP IC AIC</td>
<td>(volser,volser,...)</td>
<td>()</td>
<td>464</td>
<td>Override your installation’s MVS specification on the maximum number of volumes allowed for an output image copy data set when they are dynamically allocated.</td>
</tr>
</tbody>
</table>
### Table 96 Keywords (part 8 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDCAMS</td>
<td>BUILD</td>
<td>(input)</td>
<td>none</td>
<td>464</td>
<td>Specify SYSIN and SYSPRINT files for IDCAMS processing.</td>
</tr>
<tr>
<td></td>
<td>COPY</td>
<td>(input, output)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td>(input, output, Y)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RELOAD</td>
<td>(input, output, N)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(model)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ILDS</td>
<td>BUILD</td>
<td>(NO)</td>
<td>(NO)</td>
<td>468</td>
<td>Specify to build or initialize Indirect List Data Sets.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ONLY)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(YES)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RELOAD</td>
<td>(AUTO)</td>
<td>(AUTO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td>(BUILD)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(INIT)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMSID</td>
<td>BUILD</td>
<td>(imsID)</td>
<td>Use the IMSID from IMS generation (DFSVC000 module).</td>
<td>470</td>
<td>Specify IMSID under which reports are stored in the PDX.</td>
</tr>
<tr>
<td></td>
<td>DBSCAN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RELOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INDD</td>
<td>BUILD</td>
<td>(DFSURWF1)</td>
<td>none</td>
<td>471</td>
<td>Specify the input data set if you are not scanning the primary database. If specifying DFSURIDX, see keyword description on 471.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DFSURIDX)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INDDX</td>
<td>BUILD</td>
<td>(ddname(indxdbd),...)</td>
<td>none</td>
<td>472</td>
<td>Specify the data set to read secondary index records from.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(INDDX)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INST</td>
<td>GLBL</td>
<td>(instName)</td>
<td>('BMC SOFTWARE, INC.')</td>
<td>476</td>
<td>Specify the name of your installation as you want it to appear in the heading portion of Image Copy function reports.</td>
</tr>
<tr>
<td></td>
<td>GROUP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AIC</td>
<td>(ICPSYSIN)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>INPUT</td>
<td>REORG</td>
<td>(DB,REORG)</td>
<td>(DB,REORG)</td>
<td>473</td>
<td>Specify whether to reorganize the original database or initialize an empty database.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DB,INIT)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRLMGRP</td>
<td>REORG</td>
<td>(irlmgrp)</td>
<td>none</td>
<td>477</td>
<td>Specify the IRLM group to be used with the Snapshot Copy function.</td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDPPRINT</td>
<td>RELOAD</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>522</td>
<td>Specify the data set that will contain the Reload function reports and processing output.</td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LIM</td>
<td>Prefix</td>
<td>(value)</td>
<td>(500)</td>
<td>478</td>
<td>Notification given if orphan logical children or parents without children are found. Parameter determines how frequently to issue the notification.</td>
</tr>
<tr>
<td></td>
<td>Resolution/Update function</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>max 9999</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Table 96  Keywords (part 9 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIMITS</td>
<td>UNLOAD</td>
<td>(startValue, endValue,ddname,...)</td>
<td>none</td>
<td>479</td>
<td>Specify subsets for a parallel unload.</td>
</tr>
<tr>
<td>LOGDSN</td>
<td>REORG</td>
<td>(dsname,SINGLE)</td>
<td>none</td>
<td>483</td>
<td>Dynamically allocate the IERFDER and CRFRDER logs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(dsname,DUAL)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOGEXPDT</td>
<td>REORG</td>
<td>(yyddd)</td>
<td>none</td>
<td>485</td>
<td>Keep the logs until the specified date.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(yyyy/ddd)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOGRETD</td>
<td>REORG</td>
<td>(nmmnn)</td>
<td>none</td>
<td>485</td>
<td>Keep the logs the specified number of days.</td>
</tr>
<tr>
<td>LOGSMS</td>
<td>REORG</td>
<td>(YES,mgmt,strg,data)</td>
<td>Use the installation defaults.</td>
<td>485</td>
<td>Use SMS and the specified management, storage, and data classes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(NO)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOGSPACE</td>
<td>REORG</td>
<td>(prim,sec)</td>
<td>(100,50)</td>
<td>486</td>
<td>Specify the amount of space to allocate when dynamically allocating the logs.</td>
</tr>
<tr>
<td>LOGUNIT</td>
<td>REORG</td>
<td>(unitName,unitName)</td>
<td>none</td>
<td>486</td>
<td>Use the specified unit name(s) for the logs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LPCK</td>
<td>DBSCAN</td>
<td>(NO)</td>
<td>(YES)</td>
<td>487</td>
<td>Retrieve the logical parent concatenated key (LPCK).</td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td>(YES)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MONITOR</td>
<td>BUILD</td>
<td>(nnnnnnnn)</td>
<td>no automatic monitoring</td>
<td>490</td>
<td>Monitor job progress.</td>
</tr>
<tr>
<td></td>
<td>DBSCAN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RELOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MONUSERS</td>
<td>BUILD</td>
<td></td>
<td></td>
<td>490</td>
<td>Send monitor messages to TSO users and BMCMSG.</td>
</tr>
<tr>
<td></td>
<td>DBSCAN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RELOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAME</td>
<td>GROUP (ICPSYSIN)</td>
<td>(grpName)</td>
<td>(CMDnnn)</td>
<td>493</td>
<td>Specify a unique identifier for a group that will be used by the Image Copy function.</td>
</tr>
<tr>
<td>NEWDBD</td>
<td>UNLOAD</td>
<td>(ddname)</td>
<td>none</td>
<td>494</td>
<td>Restructure the database using this new DBD.</td>
</tr>
<tr>
<td>NOAUTH</td>
<td>REORG (Reorg function)</td>
<td>(RUN)</td>
<td>(STOP)</td>
<td>494</td>
<td>Specify the action to take if database authorization is prohibited.</td>
</tr>
</tbody>
</table>
### Table 96  Keywords (part 10 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOTIFY</td>
<td>Prefix</td>
<td>Resolution/Update function</td>
<td>(userId)</td>
<td>none</td>
<td>495  Specify TSO users to be notified when events occur within the Prefix Resolution/Update function.</td>
</tr>
<tr>
<td>OLDDBD</td>
<td>REORG</td>
<td>(ddname)</td>
<td>none</td>
<td>496  If you want to change the DBD definition during a reorganization, specify the OLDDBD definition here.</td>
<td></td>
</tr>
<tr>
<td>OSAMINIT</td>
<td>RELOAD</td>
<td>REORG GROUP</td>
<td>USERLOAD (YES) (NO) (YES)</td>
<td>499  Specify the reset of the high-used RBA for a database that has multi-volume OSAM data sets.</td>
<td></td>
</tr>
<tr>
<td>OSAMMAX</td>
<td>RELOAD</td>
<td>REORG (4) (8) (4)</td>
<td>500  Specify the size limit of an OSAM database.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTDDX</td>
<td>RELOAD</td>
<td>(ddname(indxdbd),...) (AUTO)</td>
<td>none</td>
<td>501  Specify the data set that will contain secondary index records.</td>
<td></td>
</tr>
<tr>
<td>OVFLONLY</td>
<td>RELOAD</td>
<td>REORG (segName1, segName2,..) max 25 names</td>
<td>none</td>
<td>503  Place specified segments in HDAM or PHDAM overflow area.</td>
<td></td>
</tr>
<tr>
<td>PADCHAR</td>
<td>RELOAD</td>
<td>REORG PADCHAR(x) PADCHAR('x') PADCHAR('x') PADCHAR(C'x') PADCHAR(X'xx') PADCHAR(X'00') (X'00')</td>
<td>503  Specify the character to use to pad segments that are extended between the unload and reload.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PARALLEL</td>
<td>BUILD</td>
<td>RELOAD REORG UNLOAD (YES) (NO) (YES)</td>
<td>504  Enable parallel processing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PARMBLK</td>
<td>GLBL (ICPSYSIN)</td>
<td>(modName)</td>
<td>(ICPPARMS)</td>
<td>507  Identify the name assigned to the global options module for the Image Copy function.</td>
<td></td>
</tr>
<tr>
<td>PART</td>
<td>BUILD</td>
<td>RELOAD REORG UNLOAD (ALL) (partitionName,..) (nn,nn,..) (ALL)</td>
<td>508  Perform reorganization tasks for one or more database partitions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCP</td>
<td>GLBL</td>
<td>GROUP IC AIC (ICPSYSIN) (N) (Y) (N)</td>
<td>517  Activate the interface to the Hash Checking technique of POINTER CHECKER PLUS® to verify the pointers in the database while taking an image copy of the database.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 96  Keywords (part 11 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDX</td>
<td>BUILD</td>
<td>(OKAYnnnn)</td>
<td>(OKAY0004)</td>
<td>518</td>
<td>Specify action to take if the PDX fails to open. PDX is not supported for HALDB.</td>
</tr>
<tr>
<td>PDX</td>
<td>RELOAD</td>
<td>()</td>
<td>()</td>
<td>518</td>
<td>Specify the name of the PDX data set that should be used to retrieve database specific options and to write processing statistics for the Image Copy function.</td>
</tr>
<tr>
<td>PDX</td>
<td>REORG</td>
<td>(pdxName)</td>
<td>()</td>
<td>518</td>
<td>Use a PDX to store statistical data captured during processing.</td>
</tr>
<tr>
<td>PDXPARMS</td>
<td>RELOAD</td>
<td>(pdxName)</td>
<td>()</td>
<td>520</td>
<td>Override the DBD name member in the PDX.</td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td>(pdxName)</td>
<td>()</td>
<td>520</td>
<td>Use the DBD name specified in the EXEC statement’s PARM operand.</td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td>(pdxName)</td>
<td>()</td>
<td>520</td>
<td>Specify to build primary indexes.</td>
</tr>
<tr>
<td>PRIMIND</td>
<td>BUILD</td>
<td>(YES) ~ (NO) ~ (ONLY)</td>
<td>(NO)</td>
<td>521</td>
<td>Print logical child orphans that are logical child records with no logical parent.</td>
</tr>
<tr>
<td>PRTOLC</td>
<td>REORG</td>
<td>(Y) ~ (N)</td>
<td>(Y)</td>
<td>521</td>
<td>Print logical child orphans that are logical child records with no logical parent.</td>
</tr>
<tr>
<td>PRTOLP</td>
<td>REORG</td>
<td>(Y) ~ (N)</td>
<td>(N)</td>
<td>521</td>
<td>Print logical parent orphans that are logical parent record with no logical children.</td>
</tr>
<tr>
<td>PSB</td>
<td>OREORG</td>
<td>(psbName)</td>
<td>(psbName)</td>
<td>523</td>
<td>Specify a PSB BMP for the Online Reorg function.</td>
</tr>
<tr>
<td>PTRCHECK</td>
<td>UNLOAD</td>
<td>(NO) ~ (PHY)</td>
<td>(NO)</td>
<td>524</td>
<td>Verify pointers.</td>
</tr>
</tbody>
</table>
### Table 96  Keywords (part 12 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTRERROR</td>
<td>DBSCAN UNLOAD</td>
<td>(ABEND) (ACCEPT) (ACCEPT, n)</td>
<td>(ABEND)</td>
<td>524</td>
<td>Specify the action to take if a pointer error is found.</td>
</tr>
<tr>
<td>REPORTS</td>
<td>RELOAD REORG UNLOAD</td>
<td>See Table 234 on page 526.</td>
<td>See Table 234 on page 526.</td>
<td>526</td>
<td>Generate reports.</td>
</tr>
<tr>
<td>RMBYTES</td>
<td>RELOAD REORG</td>
<td>(SEGSIZE) (MINBYTES)</td>
<td>(MINBYTES)</td>
<td>529</td>
<td>Specify how to calculate the database record size to determine when to place consecutively inserted segments into overflow.</td>
</tr>
<tr>
<td>ROUTCDE</td>
<td>GLBL (ICPSYSIN)</td>
<td>(nn,nn,...)</td>
<td>(1,7,11)</td>
<td>530</td>
<td>Specify a list of routing code numbers (1–15) that indicates which types of MVS operator consoles are to receive Image Copy function monitoring messages.</td>
</tr>
<tr>
<td>ROUTCDE</td>
<td>Prefix Resolution/Update function</td>
<td>(nn,...)</td>
<td>(1,7,11)</td>
<td>490</td>
<td>For discussion, see MVS Supervisor Macros Manual under WTO and WTOR macros.</td>
</tr>
<tr>
<td>ROUTCODE</td>
<td>BUILD COPY DBSCAN RELOAD REORG UNLOAD</td>
<td>(nn,...)</td>
<td>(2,7)</td>
<td>490</td>
<td>Specify routing codes to control destination of messages sent to the console.</td>
</tr>
<tr>
<td>SCAN (space search)</td>
<td>RELOAD REORG</td>
<td>(-nnn+mmm) (+nnn-mmm) (++nnn),(+-nnn)</td>
<td>(0) for HIDAM or PHIDAM (-10+10) for HDAM or PHDAM</td>
<td>531</td>
<td>Set the space search method.</td>
</tr>
<tr>
<td>SECINDEX</td>
<td>BUILD COPY</td>
<td>(secIndex1, secIndex2,...) (secIndexPart1, secIndexPart2,...)</td>
<td>Build all secondary indexes. (NO)</td>
<td>534</td>
<td>Specify the secondary indexes or HALDB secondary index partitions to be built.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(YES) (NO) (ONLY)</td>
<td></td>
<td></td>
<td>Specify whether to copy the secondary indexes, DBD, or both.</td>
</tr>
<tr>
<td>SEGPOS</td>
<td>RELOAD REORG</td>
<td>(segName1,n1, segName2,n2,...) max 25 names</td>
<td>none</td>
<td>535</td>
<td>Specify which segment type to place near the root.</td>
</tr>
</tbody>
</table>
### Table 96  Keywords (part 13 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGPOSO</td>
<td>RELOAD</td>
<td>ALL (segName1, segName2,...) max 25 names</td>
<td>none</td>
<td>536</td>
<td>Specify whether to place a segment in the HDAM overflow area, if it exceeds the limit specified in the SEGPOS specification.</td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SEGSEL</td>
<td>UNLOAD</td>
<td>(segName1, segName2,...) max 30 names</td>
<td>none</td>
<td>536</td>
<td>Unload only the specified segment names.</td>
</tr>
<tr>
<td>SEQERROR</td>
<td>RELOAD</td>
<td>(ABEND) (ACCEPT) (ACCEPT,n) (SAVE)</td>
<td>(ABEND)</td>
<td>537</td>
<td>Specify the action to take if a sequence error is found.</td>
</tr>
<tr>
<td></td>
<td>UNLOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHARE</td>
<td>COPY</td>
<td>(YES,UPDATE)</td>
<td>(NO)</td>
<td>540</td>
<td>Specify whether users will have update access to the database during reorganization.</td>
</tr>
<tr>
<td></td>
<td>REORG (Online Reorg function)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHARE</td>
<td>COPY</td>
<td>(YES) (NO) (YES,READ)</td>
<td>(NO)</td>
<td>540</td>
<td>Specify whether users will have read access to the database during reorganization.</td>
</tr>
<tr>
<td></td>
<td>REORG (Reorg function)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIUDSN</td>
<td>GLBL (ICPSYSIN)</td>
<td>(dataSetName)</td>
<td>(bmc.siudummy.dsn)</td>
<td>541</td>
<td>Specify a dummy data set name (1–44 characters) to use when notifying DBRC that a virtual image copy has been created.</td>
</tr>
<tr>
<td>SIUPRINT</td>
<td>BUILD</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>522</td>
<td>Specify the data set that will contain the Index Build function reports and processing output.</td>
</tr>
<tr>
<td>SIUSORT</td>
<td>BUILD</td>
<td>(I) (E)</td>
<td>(I)</td>
<td>542</td>
<td>Specify whether to sort and load indexes in separate address spaces.</td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SORTIDX</td>
<td>Prefix</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>544</td>
<td>Run Secondary Index Resolution function during Parent-Child Resolution.</td>
</tr>
<tr>
<td></td>
<td>Resolution/Update function</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SORTIN</td>
<td>Prefix</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>544</td>
<td>Create IMS-format records and read the IMS-format work file on SORTIN.</td>
</tr>
<tr>
<td></td>
<td>Resolution/Update function</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SORTWORK</td>
<td>BUILD</td>
<td>(n,ccc,unitName, dataclas,mgmtclas,storclas)</td>
<td>(3,100,SYSDA)</td>
<td>545</td>
<td>Dynamically allocate sort work data sets.</td>
</tr>
<tr>
<td></td>
<td>RELOAD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPILL</td>
<td>RELOAD</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>547</td>
<td>Specify how to store segments while evaluating the SEQERROR parm.</td>
</tr>
<tr>
<td></td>
<td>REORG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 96  Keywords (part 14 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSID</td>
<td>OREORG (Online Reorg function)</td>
<td>(imsID)</td>
<td>Use the IMSID from IMS generation (DFSVC000 module).</td>
<td>Specify the IMS subsystem to be used for the BMP to connect to the IMS control region.</td>
</tr>
<tr>
<td>STATS</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>(Y) (N)</td>
<td>(Y)</td>
<td>Specify whether to accumulate and print free space analysis information during Image Copy function processing.</td>
</tr>
<tr>
<td>SWAP</td>
<td>COPY REORG</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>Automatically swap the data set names after a successful reorganization.</td>
</tr>
<tr>
<td>SYNCTAPE</td>
<td>UNLOAD</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>Synchronize unload output tape volumes.</td>
</tr>
<tr>
<td>TIME</td>
<td>GLBL (ICPSYSIN)</td>
<td>(N) (Y)</td>
<td>(N)</td>
<td>Specify whether to provide a time stamp on Image Copy function messages in the processing log.</td>
</tr>
<tr>
<td>TRUNC</td>
<td>RELOAD REORG</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>Start each HIDAM or PHIDAM root in a new block.</td>
</tr>
<tr>
<td>UCBITMAP</td>
<td>REORG RELOAD</td>
<td>(YES) (NO)</td>
<td>(NO)</td>
<td>Specify a separate value for up to ten data set groups.</td>
</tr>
<tr>
<td>ULPPRINT</td>
<td>REORG UNLOAD</td>
<td>(ddname)</td>
<td>(BMCPRINT)</td>
<td>Specify the data set that will contain the Unload function reports and processing output.</td>
</tr>
<tr>
<td>UNIT</td>
<td>Prefix Resolution/Update function</td>
<td>(value)</td>
<td>(TAPE)</td>
<td>Specify the unit name to use when dynamically allocating the PRPURIDX work data set.</td>
</tr>
<tr>
<td>UNAUTH</td>
<td>UNLOAD</td>
<td>(N) (Y)</td>
<td>(Y)</td>
<td>Specify whether to obtain authorization for an unload.</td>
</tr>
<tr>
<td>UNLDD</td>
<td>REORG (Reorg function)</td>
<td>(ccc,n)</td>
<td>(OLD,1)</td>
<td>Specify the input ddname(s) for a reorganization using substitution characters.</td>
</tr>
<tr>
<td>UNLDDLST</td>
<td>REORG (Reorg function)</td>
<td>(ddna, ddnb)</td>
<td>none</td>
<td>Specify the input ddnames for a reorganization using assigned ddnames.</td>
</tr>
</tbody>
</table>
PLUSIN control statement syntax

The control statement data set must have the ddname PLUSIN and must contain 80-character fixed length records. A command can have keywords and comments, separated where necessary with separators and continuation characters (if statements continue to the next line). You can enter commands anywhere in positions 1 through 72 of the input statement (positions 73 through 80 are ignored). For examples of control statement syntax, see Figure 85 on page 405.

Table 96 Keywords (part 15 of 15)

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Command</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPDATE</td>
<td>Prefix Resolution/Update function</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>564</td>
<td>Run concurrent prefix update during the output phase of DFSURG30/DFSURG10. This completes the prefix update and prevents creation of a DFSURWF3 data set for updated data sets.</td>
</tr>
<tr>
<td>USEINDEX</td>
<td>DBSCAN UNLOAD</td>
<td>(YES) (dbdName) (NO)</td>
<td>(NO)</td>
<td>564</td>
<td>Unload roots through an index.</td>
</tr>
<tr>
<td>USEREXIT</td>
<td>RELOAD REORG UNLOAD</td>
<td>(exitName, YES</td>
<td>NO)</td>
<td>(exitName, U</td>
<td>L, YES</td>
</tr>
<tr>
<td>USERHDR</td>
<td>RELOAD UNLOAD</td>
<td>See Table 276 on page 571.</td>
<td>none</td>
<td>568</td>
<td>Create output records with a user-defined header (or prefix) to segment data.</td>
</tr>
<tr>
<td>USERID</td>
<td>GLBL (ICPSYSIN)</td>
<td>(userID1,...,userID2,...)</td>
<td>( )</td>
<td>573</td>
<td>Specify a list of one or more TSO users to notify when exceptional conditions occur during processing.</td>
</tr>
<tr>
<td>VIC</td>
<td>GLBL GROUP IC AIC (ICPSYSIN)</td>
<td>(N) (Y)</td>
<td>(N)</td>
<td>573</td>
<td>Create a virtual image copy of the primary and secondary indexes for the reorganized database.</td>
</tr>
<tr>
<td>XFIXED</td>
<td>Prefix Resolution/Update function</td>
<td>(Y) (N)</td>
<td>(N)</td>
<td>575</td>
<td>Allow fixed-length sort of index records.</td>
</tr>
</tbody>
</table>
Figure 85  Examples of control statement syntax

<table>
<thead>
<tr>
<th>Command</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>/* THIS LINE IS AN EXAMPLE OF A COMMENT*/</td>
<td></td>
</tr>
<tr>
<td>BUILD DBD(PRIMEDB) /* KEYWORD WITH VALUE */</td>
<td></td>
</tr>
<tr>
<td>BUILD DBD(PRIMEDB) INDD(DFSURWF1) - MONITOR(10000) /* CONTINUATION OF KEYWORDS*/</td>
<td></td>
</tr>
</tbody>
</table>

**Commands**

The control statement must begin with a valid command. A separator must follow a command.

**Keywords**

Keywords follow the command and invoke the options discussed in this chapter. Table 96 on page 390 summarizes the accepted and default values for the keywords. All keywords are non-positional. You can specify a keyword by itself, with a value following it in parentheses, or with a list following it in parentheses. If you do not specify a keyword, the utility uses the value built for that keyword as specified in “How the utilities build options” on page 385.

The keyword value can be any character string up to 255 characters. All characters are allowed, but you must enclose any character string that contains special characters in apostrophes.

**Comments**

Comments consist of an alphanumeric character string beginning with a slash-asterisk (/*) and ending with an asterisk-slash (*/). Each utility assumes a comment has ended if it reaches the end of the statement before it encounters the asterisk-slash (*/) and if the last character in the statement is not a continuation character. Comments cannot start in position 1 of an input statement.

**Separators**

Use a blank, a comma, or a comment when you need a separator. More than one separator is allowed between keywords. Do not use a separator between a keyword and its value.
Continuation characters

The valid continuation characters are the plus sign (+) and minus sign (-). Use them to continue control statements and comments that do not fit on a single line of input. The continuation character must be the last nonblank character. Use the continuation characters as follows:

- Use the plus sign (with no spaces before it) to continue values for a single keyword to a second or subsequent lines. The plus sign deletes leading separators from the continued line.

- Use the minus sign (with a space before it) to continue a list of keywords for a single command. The minus sign does not delete leading separators from the continued line.

Keyword descriptions

This section describes the global options and job-step level keywords you can set. The keywords are presented in alphabetical order. Each description includes instructions for setting the keyword through the PLUSIN or ICPSYSIN control statement data set. Each description lists which commands the keyword is valid for.

ABNDNO

The ABNDNO keyword is valid with the Prefix Resolution/Update function. You can specify the option in a Prefix Resolution/Update function global options module or in PLUSIN. Table 97 describes the keyword.

Table 97 Abend if no output records are written keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause the Prefix Resolution/Update function to abend if there are no output records written to any file.</td>
<td>ABNDNO(Y)</td>
</tr>
<tr>
<td>Allow the Prefix Resolution/Update function to continue if there are no output records written to any file.</td>
<td>ABNDNO(N)</td>
</tr>
<tr>
<td>Internal default: ABNDNO(Y)</td>
<td></td>
</tr>
</tbody>
</table>

Specify ABNDNO(Y) to cause the Prefix Resolution/Update function to abend if there are no output records written to any output file.
AGN

The AGN keyword is valid with the OREORG command. For the OREORG command, you can specify in PLUSIN. Table 98 describes the keyword.

Table 98  Application group name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the application group name for the BMP or reorganization.</td>
<td>AGN(agnName)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: none</td>
<td></td>
</tr>
</tbody>
</table>

Use the AGN keyword to provide an application group name (AGN). If you use the BMP during an Online Reorg function reorganization, use the AGN with your security product to provide security for the BMP.

ALLOWLC

The ALLOWLC keyword is valid with the REORG (Online Reorg function and Reorg function) command and the Prefix Resolution/Update function. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 99 describes the keyword.

Table 99  Allow logical children keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow logical children with no logical parents.</td>
<td>ALLOWLC(Y)</td>
</tr>
<tr>
<td>Specify to cause PRPURG20 to abend if the condition occurs.</td>
<td>ALLOWLC(N)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: ALLOWLC(N)</td>
<td></td>
</tr>
</tbody>
</table>

ALLOWLP

The ALLOWLP keyword is valid with the REORG (Online Reorg function and Reorg function) command and the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or in PLUSIN. Table 100 on page 408 describes the keyword.
Specify ALLOWLP(Y) to allow logical parents with no logical children. Specify ALLOWLP(N) to cause PRPURG20 to abend if the condition occurs.

### AMSOUT

The AMSOUT global option is valid in the Reorg function global options module (FRF$GLBL). You can also specify it in the JCL as a DD statement. You cannot specify an AMSOUT keyword in the PLUSIN control statement data set. There is no default.

When you specify the AMSOUT global option, an AMSOUT file is dynamically allocated. The AMSOUT file contains restart information. The Online Reorg function and the Reorg function use the AMSOUT file when you delay data set name swapping and when you restart post-processing tasks.

If you specify a PDS or PDSE for the AMSOUT file, the member name will be the DBD name. For best performance, BMC recommends using a PDSE.

If you want to specify an AMSOUT file in the global options module, modify the #FRFGLBL member of the sample library. Use the following syntax, and provide any valid data set name:

```plaintext
AMSOUT='dataSetName'

//AMSOUT DD DSN=dataSetName, DISP=SHR
```

You can specify an AMSOUT DD statement in the JCL as follows:

```plaintext
//AMSOUT DD DSN=dataSetName, DISP=SHR
```

The AMSOUT file is required for the Online Reorg function.

The AMSOUT file is required for the Reorg function if you want to execute the restart process to perform post-processing tasks. (You must use the restart process if post-processing tasks fail or you swap data set names manually.) DBRC must be active and the database must be registered.

#### Table 100 Allow logical parents keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow logical parents with no logical children.</td>
<td>ALLOWLP(Y)</td>
</tr>
<tr>
<td>Do not allow logical parents with no logical children.</td>
<td>ALLOWLP(N)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: ALLOWLP(C(N)</td>
<td></td>
</tr>
</tbody>
</table>

Specify ALLOWLP(Y) to allow logical parents with no logical children. Specify ALLOWLP(N) to cause PRPURG20 to abend if the condition occurs.
AOI

The AOI keyword is valid with the REORG (Reorg function) command. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 101 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the MAXM Reorg/Online automated operator interface to issue /DBD, /DBR, and /STA commands before and after a reorganization.</td>
<td>AOI(YES)</td>
</tr>
<tr>
<td>Issue /DBD, /DBR, and /STA commands manually before and after a reorganization.</td>
<td>AOI(NO)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> NO</td>
<td></td>
</tr>
</tbody>
</table>

If you have the Online Reorg function installed (even if you do not have a license for it), you can use its AOI during a Reorg function reorganization. The AOI is invoked at the beginning of the reorganization and during the data set name swapping task. Table 102 shows keywords related to the AOI keyword.

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Effect on related keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOI(YES)</td>
<td>The Reorg function forces the AUTHERR keyword to AUTHERR(WTOR)</td>
</tr>
<tr>
<td>SWAP(NO)</td>
<td>The database remains in prohibit authorization status with ACCESS=RD at the end of the reorganization.</td>
</tr>
<tr>
<td>ICP(NO)</td>
<td>The database is not started and will be left stopped with ACCESS=RD at the end of the reorganization.</td>
</tr>
</tbody>
</table>

At the beginning of the reorganization, the AOI saves the current access intent of the database, switches the intent to ACCESS=RD, and issues a /DBD command. The /DBD command places the database in read mode for the online system.
During data set name swapping, the Reorg function must have exclusive access to the database. The AOI automatically issues a /DBR command to place the database in STOPPED, NOT OPEN status. After the data set name swapping process completes successfully, the AOI automatically issues a /STA command to make the database available to users. The saved access intent attributes are restored.

**WARNING**
Regardless of the value of the SHARE keyword, the Online Reorg function and the Reorg function always require exclusive control during the data set name swapping process. If you attempt a multiple step JCL allocation, exclusive control is lost and the database becomes unavailable.

For more information, see “Automated operator interface” on page 126.

**AOIUSER**

The AOIUSER keyword is valid with the REORG (Online Reorg function) and COPY commands. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 103 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke a user exit routine before issuing a /DBR and after issuing a /STA.</td>
<td>AOIUSER(progName)</td>
</tr>
</tbody>
</table>

The AOIUSER keyword allows you to invoke a user exit from within the AOI. You might want to invoke a user exit to wait for transactions to quiesce or to allow batch jobs to complete. The AOI is invoked at the beginning of the database copy task, at the beginning of the reorganization, and during the data set name swapping task.

A sample COBOL user exit is available member AOIUXCOB of the sample library. Modify this sample as needed for your installation.

**AUTHERR**

The AUTHERR keyword is valid with the REORG (Reorg function) command. You can specify this option in a Reorg function global options module or in PLUSIN. Table 108 on page 414 describes the keyword.
The AUTHERR keyword specifies the action the Reorg function takes if DBRC authorization fails during initialization processing. DBRC authorization fails when the database is in use, whether online or in batch. If you specify AUTHERR(ABEND) and DBRC authorization fails, the job terminates with user abend code U0047. No updates are made to the database. When you resolve the reason for the authorization failure, you can restart the job.

If you specify AUTHERR(WTOR) and DBRC authorization fails, the Reorg function sends a message to the operator requesting some action. The operator can schedule a /DBR or /DBD, or cancel the batch job, and issue one of the following replies to let the Reorg function try the authorization again:

- CANCEL causes a user abend U0047.
- RETRY attempts to get the authorization from DBRC. This could lead to another operator request, if authorization fails again. After 10 unsuccessful attempts, the WTOR value automatically changes to ABEND.

The AUTHERR keyword applies only when authorization is checked during initialization processing. The Reorg function always requires exclusive control during the data set name swapping process.

If DBRC is not active, AUTHERR is ignored.

For more information, see “DBRC authorization failures” on page 105.

If you specify AOI(YES), the AUTHERR value is forced to AUTHERR(WTOR).

You can set the default value for the AUTHERR option by changing the FRF$GLBL global options module. Sample JCL is in member #FRFGLBL of the sample library.

### Table 104  DBRC authorization failure keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prompt the operator for assistance if DBRC authorization fails at initialization.</td>
<td>AUTHERR(WTOR)</td>
</tr>
<tr>
<td>Abend with user abend code U0047 if DBRC authorization fails at initialization.</td>
<td>AUTHERR(ABEND)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> ABEND</td>
<td></td>
</tr>
</tbody>
</table>

The AUTOSTOP keyword is valid with the OREORG command. You can specify this option in PLUSIN. Table 105 on page 412 describes the keyword.
The AUTOSTOP keyword specifies parameters to automatically stop the online reorganization. The correct syntax follows:

```
AUTOSTOP (RECORDS(n) TIME(hhmm) MINUTES(n) KEY(x))
```

- **RECORDS**
  
  Values range from 1 through 9999999.

- **TIME**
  
  Specify four numeric characters that represent a 24-hour clock. Values range from 0000 through 2359.

- **MINUTES**
  
  Values range from 1 through 10080 (the number of minutes in seven days).

- **KEY**
  
  Specify KEY for HIDAM databases only; KEY is not valid with MODE(RECORD) reorganizations. Specify hexadecimal or character format. Valid syntax includes:

  - `x`
  - `'x'`
  - '"x"'
  - `X'xx'
  - `C'x'`
If you specify more than one parameter, the reorganization automatically stops when the earliest parameter is triggered. For example, the following parameter triggers a reorganization to stop after moving 500,000 records or stop at 5 a.m., whichever occurs first:

```plaintext
AUTOSTOP((RECORDS,500000)(TIME,0500))
```

### BIPF

The BIPF keyword is valid with the `OREORG` and `ANALYZE` commands. You can specify this option in PLUSIN. Table 106 describes the keyword.

**Table 106  BIPF keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the expected percentage by which you want to improve database organization.</td>
<td>BIPF(n)</td>
</tr>
</tbody>
</table>

*Internal default: 25*

The BIPF keyword lets you specify the expected percentage by which to improve database record placement by block. You can specify values 1 through 99. The default value is 25. For example, specify BIPF(50) to reduce a record span from 100 blocks to 50 blocks.

Online/Defrag analyzes each record to determine the optimal improvement percentage to specify. Online/Defrag does not perform the reorganization if the specified improvement percentage cannot be accomplished.

**NOTE**

BIPF() and MBI() criteria must be met to qualify a record for reorganization. For more information, see “MBI” on page 488.

### BUFSPACE

The BUFSPACE keyword is valid with the `REORG` (Online Reorg function and Reorg function), `GROUP`, `COPY`, and `UNLOAD` commands. You can specify this option in the environment setup member (`DLIGSET0`) or in PLUSIN. Table 108 on page 414 describes the keyword.
The BUFSPACE keyword enables you to specify the maximum amount of buffer space (1 through 999 MB) that the Unload function will acquire during the unload step.

You can set the default value for the BUFSPACE option during product installation or change it at any time by changing the value of the HULUBSP option in the SETUP macro and creating an updated environment setup member (DLIGSET0). Sample JCL is in member #DLIGLBL of the sample library.

### CKUPDATE

The CKUPDATE keyword is valid with the RELOAD and REORG (Reorg function) commands. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 108 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check the RECON data set for update allocation records dated after the unload file was created. If any such records exist, terminate and issue user abend 4094.</td>
<td>CKUPDATE(YES,STOP4094)</td>
</tr>
<tr>
<td>Check the RECON data set for update allocation records dated after the unload file was created. If any such records exist, continue the job but issue return code nnnn.</td>
<td>CKUPDATE(YES,OKAYnnnn)</td>
</tr>
<tr>
<td>Do not check the RECON data set for update allocation records dated after the unload file was created.</td>
<td>CKUPDATE(NO)</td>
</tr>
</tbody>
</table>

The CKUPDATE keyword specifies whether to check the RECON data set for allocation records that have an allocation date between the date in the unload file header record and the start date of the reload.

### NOTE

The CKUPDATE keyword is valid for the Reorg function only when you specify INPUT(UNLDFILE).
If the product finds any allocation records that were created after the date of the unload file, it ends (STOP) or continues (OKAY) and issues the abend or return code you specify. The product issues message BMC90089I for each allocation record that is created between the unload and the reload.

The CKUPDATE keyword is honored only when all of the following conditions are met:

- DBRC is active.
- The database is registered to DBRC.
- The unload file was created by the Unload function. (The IMS HD Unload utility does not create a time stamp in the unload file header record.)

**COMPRESS, EXPAND**

The COMPRESS keyword is valid with the RELOAD command. The EXPAND keyword is valid with the REORG (Online Reorg function and Reorg function) and UNLOAD commands. You can specify the COMPRESS option in a Reload function global options module or in PLUSIN. You can specify the EXPAND option in an Unload function or Reorg function global options module or in PLUSIN; however, the Reorg function ignores the EXPAND specification in a global options module. Table 109 describes the keywords.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load segments already compressed (bypass compression routine). (Reload function)</td>
<td>COMPRESS(NO)</td>
</tr>
<tr>
<td>Compress segments as they are loaded (invoke compression routine). (Reload function)</td>
<td>COMPRESS(YES)</td>
</tr>
<tr>
<td>Unload segments in compressed format (do not expand). (Unload function)</td>
<td>EXPAND(NO)</td>
</tr>
<tr>
<td>Expand segments as they are unloaded. (Unload function)</td>
<td>EXPAND(YES)</td>
</tr>
<tr>
<td>Expand the segments when unloading and compress when reloading. (Reorg function)</td>
<td>EXPAND(YES)</td>
</tr>
<tr>
<td>Do not expand the segments when unloading and do not compress when reloading. (Reorg function)</td>
<td>EXPAND(NO)</td>
</tr>
</tbody>
</table>

**Internal default:**
- Reload function: COMPRESS(YES)
- Unload function: EXPAND(YES)
- Reorg function: EXPAND(NO)
- Online Reorg function: EXPAND(NO)
Unload function and Reload function

For databases that use the segment edit/compression exit, use the EXPAND keyword to control whether the Unload function expands the segments or places them into the HD Unload data set in compressed format.

For faster unload and reload, unload the segments in compressed format instead of expanded format. If you are using the IBM-provided reload utilities, unloading compressed segments causes the following problems:

- HD Unload data sets containing segments in compressed format require special DBDs for the IBM-provided reload utilities.

- If the source segment for a secondary index was unloaded in compressed format, a reload with the IBM-provided reload utilities cannot be successful (even with a special DBD).

If you use the Reload function to reload, all compressed segment types can remain compressed, including source segments for a secondary index. The Reload function expands any compressed secondary index source segments to build secondary indexes.

In most cases, the Reload function can automatically detect whether the segments are compressed or expanded. If you use the IBM HD Sort utility to sort the HD Unload data set before the reload, you must ensure that you coded the correct value for the Reload function COMPRESS keyword. The IBM HD Sort utility deletes the special header record that contains the flag indicating whether the data is compressed or expanded. The BMC HD Sort utility does not delete the special header record. For more information, see “HD Sort utility” on page 227.

**WARNING**

If you changed the DBD to implement compression for a segment type and other segment types in the DBD already have compression implemented, you must specify expansion and unload with the old DBD before reloading with the changed DBD.

This warning also applies to changes (adding length to a segment) to an existing segment that already has compression defined. You must specify expansion and unload with the old DBD before reloading with the changed DBD.

The Reload function API ignores the COMPRESS option. During the initial load of the database, the compression routine is always invoked if it is specified for a segment. For more information, see “API” on page 234.
Compressed segments in the HD Unload file

If the Unload function unloaded the segments in compressed format, the Reload function will not doubly compress them, regardless of the value you specify for this keyword. See Table 110.

Table 110  Compression: unload data set created by Unload function

<table>
<thead>
<tr>
<th>Reload function keyword</th>
<th>Unload function keyword EXPAND(NO)</th>
<th>Unload function keyword EXPAND(YES)</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPRESS(NO)</td>
<td>The Unload function does not expand data; the Reload function does not compress data.</td>
<td>The Unload function expands data; the Reload function compresses data.</td>
</tr>
<tr>
<td>COMPRESS(YES)</td>
<td>The Unload function does not expand data; the Reload function does not compress data.</td>
<td>The Unload function expands data; the Reload function compresses data.</td>
</tr>
</tbody>
</table>

(DefaultValue)

If a program other than the Unload function unloaded the segments in compressed format, specify COMPRESS(NO). Otherwise, the segments might be compressed twice. See Table 111.

### NOTE

If the BMC DATA PACKER®/IMS product was used for compression, it will check whether a segment is compressed, to avoid a segment being doubly compressed.

Table 111  Compression: unload data set not created by Unload function

<table>
<thead>
<tr>
<th>Reload function keyword</th>
<th>Compressed data in HD Unload data set</th>
<th>Expanded data in HD Unload data set</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPRESS(NO)</td>
<td>Okay</td>
<td>The Reload function does not compress data. This creates an invalid IMS database.</td>
</tr>
<tr>
<td>COMPRESS(YES)</td>
<td>The Reload function might double compress data (cyclical compression). This creates an invalid IMS database.</td>
<td>Okay</td>
</tr>
</tbody>
</table>

(DefaultValue)

You can set the default value for the COMPRESS and EXPAND options by using the Reload function and Unload function ISPF panels or the equivalent batch jobs. Sample JCL is in member #DBUGLBL of the sample library.

### Online Reorg function and Reorg function

The EXPAND keyword determines whether compressed segments are expanded as they are unloaded.
The EXPAND keyword is applicable only if the database uses compression or if you are adding compression during the reorganization.

If compression is in effect for the database, the following information is true for the Reorg function and the Online Reorg function:

- If you specify EXPAND(YES), the Reorg function and the Online Reorg function expand the data during the unload process and compress the data during the reload process.

- If you specify EXPAND(NO), the Reorg function and the Online Reorg function do not expand the data during the unload process and do not compress the data when it is reloaded.

A COMPRESS keyword is not available in the Reorg function or the Online Reorg function. If the Reorg function expands data being unloaded, it compresses the data when it is reloaded. If the Reorg function does not expand data being unloaded, it does not compress data being reloaded.

If you are using the OLDDBD keyword and either of the following conditions exists, you must specify EXPAND(YES) in the Reorg function or the Online Reorg function:

- You are changing a database that has compression, and the data must be expanded (for example, if you are changing a compression routine).

- You are adding compression to or removing compression from a database.

If you are using the OLDDBD keyword and you are adding compression to a database that previously had no compression, EXPAND(NO) is ignored.

The Reorg function and the Online Reorg function always expand compressed secondary index source segments so that the secondary index can be loaded successfully.

The Reorg function and the Online Reorg function ignore the EXPAND keyword if both the old DBD and the new DBD do not have compression.

The EXPAND keyword is not valid when you specify INPUT(UNLDFILE) in the Reorg function.
CREORG keyword is valid with the UNLOAD and REORG commands to support the conditional reorganization feature in MAXM Database Advisor for IMS. The keyword specifies the action that the utility takes if MAXM Database Advisor for IMS is unavailable. You can specify this option in PLUSIN. Table 112 describes the keyword.

**Table 112  Conditional reorganization keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the conditional reorganization feature. If MAXM Database Advisor for IMS</td>
<td>CREORG(YES,OKAYnnnn)</td>
</tr>
<tr>
<td>is not available, proceed with the reorganization and end with a condition code nnnn.</td>
<td></td>
</tr>
<tr>
<td>Use the conditional reorganization feature. If MAXM Database Advisor for IMS</td>
<td>CREORG(YES,STOPnnnn)</td>
</tr>
<tr>
<td>is not available, abort the reorganization and end with a user abend code nnnn.</td>
<td></td>
</tr>
<tr>
<td>Do not use the conditional reorganization feature.</td>
<td>CREORG(NO)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> CREORG(YES,OKAY0004)</td>
<td></td>
</tr>
</tbody>
</table>

When MAXM Database Advisor for IMS is installed and active, the Reorg function queries MAXM Database Advisor for IMS for the reorganization status of each database scheduled for reorganization. If the status indicates that the reorganization is unnecessary, the Reorg function issues an informational message to bypass the reorganization. The job ends with the condition code or abend code specified by the user in the conditional reorganization settings.

**Multiple database reorganizations**

If none of the databases in the group need to be reorganized, the reorganizations are bypassed. For databases that do not have logical relationships, only the databases that need to be reorganized are reorganized.

If the databases have logical relationships and at least one database needs to be reorganized, all databases in the group are reorganized.

An informational message is issued for each database that did not need to be reorganized, and the job ends with the highest user-supplied return code or abend code.

**HALDBs**

If any of the selected partitions needs to be reorganized, all selected partitions are reorganized.
DBD

The DBD keyword is valid with the GROUP, COPY, BUILD, IC, AIC, OREORG, and ANALYZE commands. This keyword is available in PLUSIN and required in ICPSYSIN; it is not available as a global option. Table 113 describes the keyword.

Table 113 Primary DBD name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN or ICPSYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the DBD name for the primary database.</td>
<td>DBD(dbdName)</td>
</tr>
<tr>
<td>Internal default: none</td>
<td></td>
</tr>
<tr>
<td>(for the ANALYZE command, the default value is the DBD name of the file processed by the Extractor and Recommendation Selector)</td>
<td></td>
</tr>
</tbody>
</table>

The DBD keyword specifies the name of the DBD for the primary database.

For the GROUP command, the DBD keyword is required. Although BMC supports multiple COPY statements in a single job step, the DBD on each COPY statement must be unique. You cannot have two COPY statements that specify the same DBD.

For the BUILD command, you must specify at least one DBD name. You can specify as many as 20 DBD names if you are using a DFSURWF1 file as input. If you are scanning the database, you can specify only one DBD name.

For the IC and AIC command, you must specify only one DBD name per command.

The DBD keyword is required with the OREORG command.

The DBD keyword is optional with the ANALYZE command. If you use the Online/Defrag Extractor and Recommendation Selector or the Online/Defrag Analyzer, the DBD name is automatically passed. If you specify a DBD, it must match the DBD that was passed. The job abends if the DBDs are not an exact match. The primary purpose of specifying a DBD is to verify the name of the DBD.

DBDSGRP

The DBDSGRP keyword is optional on the GROUP command. This keyword is available in ICPSYSIN; it is not available as a global option. Table 114 on page 421 describes the keyword.
Use the DBDSGRP keyword to identify the name of a valid database data set group recorded in DBRC. When you specify the DBDSGRP keyword, the Image Copy function generates image copy data sets for all data set groups defined in the database data set group. For more information, see “GROUP control statement” on page 335.

If you use the DBDSGRP keyword, do not include the NAME keyword.

You can specify VIC(Y) in conjunction with the DBDSGRP keyword to build a virtual image copy of all indexes (primary and secondary) defined as part of the database data set group. Indexes that are not defined to the database data set group, but that have the primary database defined, are also processed. For more information, see “Virtual image copies” on page 330.

### Table 114 Image copy database data set group name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate image copy data sets for all data set groups defined in the database data set group.</td>
<td>DBDSGRP(dbdsName)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: none</td>
<td></td>
</tr>
</tbody>
</table>

Use the DBDSGRP keyword to identify the name of a valid database data set group recorded in DBRC. When you specify the DBDSGRP keyword, the Image Copy function generates image copy data sets for all data set groups defined in the database data set group. For more information, see “GROUP control statement” on page 335.

If you use the DBDSGRP keyword, do not include the NAME keyword.

You can specify VIC(Y) in conjunction with the DBDSGRP keyword to build a virtual image copy of all indexes (primary and secondary) defined as part of the database data set group. Indexes that are not defined to the database data set group, but that have the primary database defined, are also processed. For more information, see “Virtual image copies” on page 330.

### Table 115 Log switch keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switch the online log data sets (OLDS) during the /DBR at the end of the reorganization.</td>
<td>DBR(FEOV)</td>
</tr>
<tr>
<td>Do not switch the OLDS during the /DBR at the end of the reorganization.</td>
<td>DBR(NOFOEOV)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: DBR(FEOV)</td>
<td></td>
</tr>
</tbody>
</table>

The DBR keyword is valid with the REORG (Online Reorg function) command. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 115 describes the keyword.

The DBR keyword specifies whether to force end-of-volume on the OLDS for the /DBR at the end of the reorganization.

When you specify DBR(FEOV), or accept it as the default, the Online Reorg function issues a /SWI OLDS command to IMS. The FEOV parameter instructs IMS to switch the online log data sets. DBR(FEOV) behaves exactly the same as the IMS command /DBR DB dbdName.
If you specify DBR(NOFOEOV), the online log data sets are not switched. DBR(NOFOEOV) behaves exactly the same as the IMS command /DBR DB dbdName NOFOEOV. Not switching the online log data sets affects recovery time. The Online Reorg function created a batch log, whose time stamps overlap those of an SLDS. If you must recover the database, change accumulation (log merge) is required; this extends the length of the recovery time. However, if you use RECOVERY PLUS for IMS to recover the database, you do not need to run the change accumulation.

For more information, see “DBR commands” on page 128.

DBR(NOFOEOV) is not allowed for IMS 4.1 and earlier.

### DBRC (Index Build and Reorg functions)

The DBRC keyword is valid with the BUILD and REORG commands. You can specify this option in a global options module or in PLUSIN. Table 116 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke DBRC.</td>
<td>DBRC(YES)</td>
</tr>
<tr>
<td>Do not invoke DBRC.</td>
<td>DBRC(NO)</td>
</tr>
<tr>
<td>Use your installation’s default for invoking DBRC.</td>
<td>DBRC() or do not specify the DBRC keyword</td>
</tr>
</tbody>
</table>

**Internal default:** Use the DBRC specification in the IMSCNTL macro.

The DBRC keyword specifies whether to invoke DBRC. DBRC(YES) is required for HALDBs.

**Table 117** shows keywords that are related to the DBRC keyword.

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Effect on related keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICNEEDED</td>
<td>The ICNEEDED keyword becomes effective when you specify DBRC(YES) or when it is your installation default. For more information, see “ICNEEDED” on page 457.</td>
</tr>
<tr>
<td>DYNALLOC(Y)</td>
<td>You cannot specify DBRC(NO) if you specify DYNALLOC(YES).</td>
</tr>
</tbody>
</table>

If DBRC=FORCE was specified during IMS system generation and you specify DBRC(NO) with the BUILD command, the function terminates with a U0047 abend.
To use your IMS system default for invoking DBRC, do not specify the DBRC keyword.

You can set the default value for the DBRC option by using the Index Build function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

**DBRC (Prefix Resolution/Update function)**

The DBRC keyword is valid with the Prefix Resolution/Update function. This option applies only when a DBRC-sensitive program (such as DFSURGP0) is being run without the region controller. Table 118 shows how the DBRC keyword works.

<table>
<thead>
<tr>
<th>PRP global option</th>
<th>IMS GEN stage 1 option</th>
<th>DBRC used?</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBRC(Y)</td>
<td>DBRC=FORCE</td>
<td>yes</td>
</tr>
<tr>
<td>DBRC(Y)</td>
<td>DBRC=YES</td>
<td>yes</td>
</tr>
<tr>
<td>DBRC(Y)</td>
<td>DBRC=NO</td>
<td>no</td>
</tr>
<tr>
<td>DBRC(N)</td>
<td>DBRC=FORCE</td>
<td>yes</td>
</tr>
<tr>
<td>DBRC(N)</td>
<td>DBRC=YES</td>
<td>no</td>
</tr>
<tr>
<td>DBRC(N)</td>
<td>DBRC=NO</td>
<td>no</td>
</tr>
</tbody>
</table>

If you are running concurrent prefix update (UPDATE(Y)) and use DBRC to handle database recovery, you must specify DBRC(Y) at the global level (on the PLUSIN GLBL statement or in the global options module).

**DDN**

The DDN keyword is optional on the IC and AIC commands. This keyword is available in ICPSYSIN; it is not available as a global option. Table 119 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the ddname associated with the data set group to be processed.</td>
<td>DDN(ddname)</td>
</tr>
</tbody>
</table>

**Internal Default:** none
If you omit the DDN keyword, the Image Copy function implicitly processes each data set group of the named DBD.

**DESC (Image Copy function)**

The DESC keyword is valid with the GLBL command. This keyword is available in ICPSYSIN; it is not available as a global option. Table 120 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the descriptor code numbers (1–11) that designate the manner in which MVS is to manage the highlighting and retention of monitoring messages issued to MVS consoles.</td>
<td>DESC(nn,nn,...)</td>
</tr>
</tbody>
</table>

**Internal default:** DESC(7)

Specify a string of up to 20 characters to identify the descriptor code numbers (1–11). Separate the numbers with commas. Commas count as one character.

This keyword works with the ROUTCDE keyword.

For more information, see “Monitoring and reporting options” on page 334. You can also refer to the description of the IBM WTO macro in the IBM *Supervisor Services and Macro Instructions* manual or check with your MVS systems programmer.

**DFSDBUX1**

The DFSDBUX1 keyword is valid with the UNLOAD, RELOAD, REORG (Reorg function), and GROUP commands. You can specify this keyword in the environment setup member (DLIGSET0) or in PLUSIN. Table 121 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke the DFSDBUX1 user exit (Unload function and Reload function).</td>
<td>DFSDBUX1(YES)</td>
</tr>
<tr>
<td>Do not invoke the DFSDBUX1 user exit (Unload function and Reload function).</td>
<td>DFSDBUX1(NO)</td>
</tr>
<tr>
<td>Invoke the DFSDBUX1 user exit during unload processing but not during reload processing (Reorg function).</td>
<td>DFSDBUX1(YES,NO)</td>
</tr>
</tbody>
</table>
Chapter 14 User-controlled options

The DFSDBUX1 keyword specifies whether to invoke the DFSDBUX1 user exit, which is available in the IBM IMS/ESA Year 2000 Exit Tool product. For more information about the user exit, see the appropriate IBM publication.

The DFSDBUX1 keyword is valid only if you have an appropriate translation table. If you do not have a translation table, the keyword is ignored. When you create the translation table, the user exit is automatically invoked when you unload or reload data. If you do not want the user exit to be invoked, you must specify DFSDBUX1(NO).

The values for the DFSDBUX1 keyword must match during unload and reload. The only time that you can specify different values is before you create the translation table. For example, the first time you use the exit, you could unload the data in its original format, create the translation table, and reload the data in converted format.

If you are implementing or changing a DFSDBUX1 exit between the unload and the reload and if the DFSDBUX1 exit will change the root key, you must specify HDSORT(YES) or run the HD Sort utility between the unload and the reload.

### Table 121 DFSDBUX1 keyword (part 2 of 2)

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do not invoke the DFSDBUX1 user exit during unload processing but invoke it during reload processing (Reorg function).</td>
<td>DFSDBUX1(NO,YES)</td>
</tr>
<tr>
<td>Invoke the DFSDBUX1 user exit during unload and reload processing (Reorg function).</td>
<td>DFSDBUX1(YES,YES)</td>
</tr>
<tr>
<td>Do not invoke the DFSDBUX1 user exit (Reorg function).</td>
<td>DFSDBUX1(NO,NO)</td>
</tr>
</tbody>
</table>

**Internal Default:**
- Unload function: DFSDBUX1(YES)
- Reload function: DFSDBUX1(YES)
- Reorg function: DFSDBUX1(YES,YES)

The DFSURGU1 keyword specifies whether to invoke the DFSDBUX1 user exit, which is available in the IBM IMS/ESA Year 2000 Exit Tool product. For more information about the user exit, see the appropriate IBM publication.

The DFSURGU1 keyword is valid with the UNLOAD command. You can specify this option in an Unload function global options module or in PLUSIN. Table 122 describes the keyword.

### Table 122 HD Unload record format keyword (part 1 of 2)

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write the standard IMS (unabbreviated) HD Unload record prefixes.</td>
<td>DFSURGU1(LONG)</td>
</tr>
<tr>
<td>Write abbreviated HD Unload record prefixes.</td>
<td>DFSURGU1(SHORT)</td>
</tr>
<tr>
<td>Write super-abbreviated HD Unload record prefixes.</td>
<td>DFSURGU1(XSHORT)</td>
</tr>
</tbody>
</table>
The standard IMS unload data set contains one record for each database segment unloaded, plus a header and trailer record. Each record consists of a prefix (not the segment’s prefix) and segment data. If the database does not contain any logical relationships, you can use the DFSURGU1 keyword to tell the Unload function to write the HD Unload records with a short record prefix. The short prefix (SHORT) is 20 bytes shorter than the standard version. The extra-short prefix (XSHORT) is 30 bytes shorter than the standard version.

If you specify short prefixes and the database has logical relationships, the Unload function overrides the DFSURGU1 keyword and creates the standard HD Unload records. If you request dual HD Unload output with a DFSURGU2 DD statement, the second data set will contain HD Unload records in the same format as those in the DFSURGU1 data set.

**Restrictions**

The short records (SHORT and XSHORT) are acceptable only to the Reload function. Short records make the Unload function and the Reload function run faster.

The user format records (FRMT1, FRMT2, and FRMT3) are not valid if you specify HDSORT(YES) or HDSORT(AUTO) in the reload job.

You cannot use the Unload function USERHDR keyword with the DFSURGU1 keyword.

**WARNING**

If you specify SHORT or XSHORT for the DFSURGU1 keyword, do not sort the output HD Unload data set with the IBM Physical Sequential Sort/Reload (PSSR) utility. This utility will not handle the short records correctly. Use the HD Sort utility to sort the short records correctly. For more information, see “HD Sort utility” on page 227.

You can set the default value for the DFSURGU1 option by using the Unload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library. If the unload function is used as input to a load, only the LONG option is supported for PHDAM and PHIDAM databases. Other formats may be used to pass the unload file to user applications.
Standard and short format records

You can specify the IMS-compatible format (LONG), a short format (SHORT), or an extra-short format (XSHORT) for segment record prefixes. Segment records (with the prefix formats you specify), header records, and trailer records are written to the unload file. The prefixes you specify are not contained in the header and trailer records. The segment code field on the header and trailer records is always X’00’.

IMS-compatible record format (LONG)

When you specify DFSURGU1(LONG), the Unload function writes a variable-length record for each retrieved database segment containing the data fields. For full function databases, the unload record format is described in the DFSURGUF macro in your GENLIBB. For HALDB partitioned databases, the unload record format is described in the DFSURGUP.

Short record format (SHORT)

When you specify DFSURGU1(SHORT), the Unload function writes a variable-length record for each retrieved database segment containing the data fields shown in Table 123.

Table 123  Short (SHORT) records

<table>
<thead>
<tr>
<th>RGUDSECT</th>
<th>Dec</th>
<th>Hex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGULEN</td>
<td>DC</td>
<td>H’0’</td>
<td>0 0</td>
</tr>
<tr>
<td>RGUZZ</td>
<td>DC</td>
<td>H’0’</td>
<td>2 2</td>
</tr>
<tr>
<td>RGUSEGLV</td>
<td>DC</td>
<td>X’00’</td>
<td>4 4</td>
</tr>
<tr>
<td>RGUHSDF</td>
<td>DC</td>
<td>X’00’</td>
<td>5 5</td>
</tr>
<tr>
<td>RGUHDLRN</td>
<td>DC</td>
<td>H’0’</td>
<td>6 6</td>
</tr>
<tr>
<td>RGUSEGLN</td>
<td>DC</td>
<td>H’0’</td>
<td>8 8</td>
</tr>
<tr>
<td>RGUSEGNM</td>
<td>DC</td>
<td>CL8’</td>
<td>10</td>
</tr>
<tr>
<td>RGUSEGDF</td>
<td>DC</td>
<td>X’00’</td>
<td>18</td>
</tr>
<tr>
<td>SEGMENT</td>
<td>EQU</td>
<td>*</td>
<td>19</td>
</tr>
</tbody>
</table>

Extra-short record format (XSHORT)

When you specify DFSURGU1(XSHORT), the Unload function writes a variable-length record for each retrieved database segment containing the data fields shown in Table 124 on page 428.
User format records

Specify DFSURGU1(FRMTn), where \( n \) is 1, 2, or 3, to produce an output data set for subsequent processing in the user formats described in the following sections. These user formats are compatible with the IBM High Speed Sequential Reload (HSSR) utility.

If you specify DFSURGU1(FRMTn) and you want to reload the records with the Reload function, you must specify the Reload function USERHDR keyword and name the individual items composing the header record.

User output record format 1 (FRMT1)

When you specify DFSURGU1(FRMT1), the Unload function writes a variable-length record for each retrieved database segment containing the data fields shown in Table 125.

<table>
<thead>
<tr>
<th>Table 124 Extra-short (XSHORT) records</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RGU DSECT</strong></td>
</tr>
<tr>
<td>RGULEN</td>
</tr>
<tr>
<td>RGUZZ</td>
</tr>
<tr>
<td>RGUSEGLV</td>
</tr>
<tr>
<td>RGUSEGDF</td>
</tr>
<tr>
<td>RGUSEGLN</td>
</tr>
<tr>
<td>SEGMENT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 125 User format 1 (FRMT1) records</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>REC1 DSECT</strong></td>
</tr>
<tr>
<td>REC1LEN</td>
</tr>
<tr>
<td>REC1ZZ</td>
</tr>
<tr>
<td>REC1SC</td>
</tr>
<tr>
<td>REC1LEV</td>
</tr>
<tr>
<td>REC1DATA</td>
</tr>
</tbody>
</table>
User output record format 2 (FRMT2)

When you specify DFSURGU1(FRMT2), the Unload function writes a variable-length record for each retrieved database segment containing the data fields shown in Table 126.

Table 126  User format 2 (FRMT2) records

<table>
<thead>
<tr>
<th>REC2 DSECT</th>
<th>Dec</th>
<th>Hex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REC2LEN</td>
<td>DC</td>
<td>H'0' 0 0</td>
<td>OS record length field</td>
</tr>
<tr>
<td>REC2ZZ</td>
<td>DC</td>
<td>H'0' 2 2</td>
<td>ZZ (reserved for OS)</td>
</tr>
<tr>
<td>REC2SC</td>
<td>DC</td>
<td>X'00' 4 4</td>
<td>segment code</td>
</tr>
<tr>
<td>REC2LEV</td>
<td>DC</td>
<td>X'00' 5 5</td>
<td>segment level</td>
</tr>
<tr>
<td>REC2SYM</td>
<td>DC</td>
<td>CL8' ' 6 6</td>
<td>symbolic segment name</td>
</tr>
<tr>
<td>REC2IOAL</td>
<td>DC</td>
<td>H'0' 14 E</td>
<td>length of the segment as returned by the Unload function</td>
</tr>
<tr>
<td>REC2KOFs</td>
<td>DC</td>
<td>H'0' 16 10</td>
<td>offset of the sequence field within the segment data (zero if the segment has no sequence field)</td>
</tr>
<tr>
<td>REC2KEYL</td>
<td>DC</td>
<td>H'0' 18 12</td>
<td>length of the sequence field (zero is the segment has no sequence field)</td>
</tr>
<tr>
<td>REC2MKL</td>
<td>DC</td>
<td>H'0' 20 14</td>
<td>a field containing zeros for compatibility with the *F3 format. In this field, the *F3 format contains the actual length of the concatenated PCB key feedback area.</td>
</tr>
<tr>
<td>REC2DATA</td>
<td>EQU</td>
<td>* 22 16</td>
<td>segment data as returned by the Unload function and as seen by the subsequent program</td>
</tr>
</tbody>
</table>

User output record format 3 (FRMT3)

When you specify DFSURGU1(FRMT3), the Unload function writes a variable-length record for each retrieved database segment containing the data fields shown in Table 127.

Table 127  User format 3 (FRMT3) records (part 1 of 2)

<table>
<thead>
<tr>
<th>REC3 DSECT</th>
<th>Dec</th>
<th>Hex</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REC3LEN</td>
<td>DC</td>
<td>H'0' 0 0</td>
<td>OS record length field</td>
</tr>
<tr>
<td>REC3ZZ</td>
<td>DC</td>
<td>H'0' 2 2</td>
<td>ZZ (reserved for OS)</td>
</tr>
<tr>
<td>REC3SC</td>
<td>DC</td>
<td>X'00' 4 4</td>
<td>segment code</td>
</tr>
<tr>
<td>REC3LEV</td>
<td>DC</td>
<td>X'00' 5 5</td>
<td>segment level</td>
</tr>
<tr>
<td>REC3SYM</td>
<td>DC</td>
<td>CL8' ' 6 6</td>
<td>symbolic segment name</td>
</tr>
<tr>
<td>REC3IOAL</td>
<td>DC</td>
<td>H'0' 14 E</td>
<td>length of the segment as returned by the Unload function</td>
</tr>
<tr>
<td>REC3KOFs</td>
<td>DC</td>
<td>H'0' 16 10</td>
<td>offset of the sequence field within the segment data (zero if the segment has no sequence field)</td>
</tr>
</tbody>
</table>
The DYNALLOC keyword is valid with the BUILD, DBSCAN, RELOAD, REORG, GROUP, COPY, UNLOAD, and OREORG commands. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 128 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamically allocate the database data sets using the data set names specified in the RECON data sets.</td>
<td>DYNALLOC(YES)</td>
</tr>
<tr>
<td>Do not dynamically allocate the database data sets.</td>
<td>DYNALLOC(NO)</td>
</tr>
<tr>
<td>Dynamically allocate all database data sets using the data set names specified in the RECON data sets, and dynamically allocate output (shadow) database data sets with the data set name extension .Z (Online Reorg function and Reorg function).</td>
<td>DYNALLOC(YES,YES,Z)</td>
</tr>
<tr>
<td>Dynamically allocate the output database data sets (valid only when INPUT(UNLDFILE) is specified in Reorg function).</td>
<td>DYNALLOC(NO,YES)</td>
</tr>
<tr>
<td>Use only when INDD/OUTDD is not specified. Valid for full function databases and HALDBs.</td>
<td>DYNALLOC(N,Y,Z)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> NO</td>
<td></td>
</tr>
</tbody>
</table>

The DYNALLOC keyword allows MAXM Reorg solutions to perform dynamic allocation as follows:

- **Unload function**—dynamically allocate the database data sets for the database being unloaded or scanned
- **Reload function**—dynamically allocate the database data sets for the database being loaded
Index Build function—dynamically allocate database data sets for the database being scanned and the indexes being built

Online Reorg function and Reorg function—dynamically allocate all necessary database data sets

When you specify the DYNALLOC keyword, DBRC must be active and all databases (including the index databases) must be registered in the RECON data sets. See “DBRC (Index Build and Reorg functions)” on page 422. MAXM Reorg solutions use the data set names specified in the RECON data sets when allocating the database data sets.

If you do not specify DYNALLOC keyword or if you specify DYNALLOC(NO), you must include the database data sets in your JCL.

**NOTE**

For VSAM data sets, if you do not have the target data sets (Reload function and Index Build function) or the shadow data sets (Online Reorg function or Reorg function) preallocated, you must specify an IDCAMS keyword. The IDCAMS keyword tells the product to “create” the data sets before dynamic allocation.

### Shadow data sets

For full function databases, shadow data set names are identified by the last qualifier in the data set name. For HALDBs, shadow data set names are identified by the next to last qualifier in the data set name.

### Online Reorg function and Reorg function

When you request dynamic allocation with the DYNALLOC keyword, DBRC is required, and the database data sets must be registered in the RECON data sets. If a corresponding DD statement is in the JCL, the utility compares it to the name specified in the RECON data sets. If the data set names do not match, processing fails.

Set the values for the DYNALLOC keyword as shown in Table 129.

| Table 129  DYNALLOC keyword and other keywords (part 1 of 2) |
|------------------------|------------------------|------------------------|
| When you specify        | Function                | Explanation             |
| DYNALLOC(YES)           | Online Reorg function   | dynamically allocates the input database data sets |
|                        | Reorg function          |                         |
| DYNALLOC(YES) and SWAP(x,JC) | Reorg function         | dynamically allocates the input database data sets and the output secondary index data sets |
If you do not specify DYNALLOC, include the database DD statements as follows:

- For the Online Reorg function, the following applies:
  - Include the output (reorganized) database DD statements in the JCL, using the ddnames in the DBD. Do not include the original database DD statements.
  - For OSAM databases, you cannot specify DYNALLOC(NO,x).

- For the Reorg function, the following applies:
  - Include the database DD statements using UNLDD or UNLDDLST. For more information, see “UNLDD” on page 560 and “UNLDDLST” on page 562.
  - You can use the original data set names for the secondary index output data set names. If you specify any secondary index data set names, you must specify all of them.

DYNALLOC(YES) is inherent for HALDBs. Dynamic allocation is always used with HALDB reorganizations. HALDB shadow data set names are comprised of a prefix and suffix. The database administrator provides the prefix when defining the HALDB. DBRC generates the suffix. BMC follows the IBM standard to construct the data set name: .Z is appended to the prefix and then the suffix is added (prefix.Z.suffix).

---

### Table 129 DYNALLOC keyword and other keywords (part 2 of 2)

<table>
<thead>
<tr>
<th>When you specify</th>
<th>Function</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>DYNALLOC(YES,YES,x)</td>
<td>Online Reorg function</td>
<td>dynamically allocates the input and output database data sets and appends a suffix to the data set names. (The IDCAMS keyword is required if the shadow data sets do not exist.) For example, your original (disorganized) data set is named bmc.test.x1234. When you specify DYNALLOC(YES,YES), the output (reorganized) data set is named bmc.test.x1234.Z (Z is the default). When you specify DYNALLOC(YES,YES,K), the output data set is named bmc.test.x1234.K. You can specify any alphabetic character except T as a suffix.</td>
</tr>
<tr>
<td>DYNALLOC(YES,YES,x)</td>
<td>Reorg function</td>
<td>is not valid when you specify SWAP(x,IC)</td>
</tr>
<tr>
<td>DYNALLOC(YES) or DYNALLOC(YES,YES,x)</td>
<td>Reorg function</td>
<td>requires the UNLDD or UNLDDLST keyword. When you use the UNLDD keyword, or accept it as the default, ensure that the UNLDD keyword value makes the ddname unique (the default is UNLDD(OLD,1)).</td>
</tr>
<tr>
<td>DYNALLOC(NO,YES)</td>
<td>Reorg function</td>
<td>valid only when you specify INPUT(UNLDFILE). Otherwise, you cannot dynamically allocate only the output data sets.</td>
</tr>
</tbody>
</table>
Online/Defrag function

The DYNALLOC keyword allows Online/Defrag to dynamically allocate the database data sets for the database being reorganized.

When you specify DYNALLOC(YES), DBRC must be active and all primary database data sets must be registered in the RECON data sets.

If you do not specify the DYNALLOC keyword or if you specify DYNALLOC(NO), you must include the primary database data sets in the JCL.

EOJACCES

The End of Job Access (EOJACCES) keyword is valid with the REORG (Online Reorg function and Reorg function) command. You can specify this keyword in the environment setup member (DLIGSET0) or in PLUSIN. Table 130 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this is PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue a /STA db ACCESS=EX after the reorg.</td>
<td>EOJACCES(EX)</td>
</tr>
<tr>
<td>Issue a /STA db ACCESS=UP after the reorg.</td>
<td>EOJACCES(UP)</td>
</tr>
<tr>
<td>Issue a /STA db ACCESS=RD after the reorg.</td>
<td>EOJACCES(RD)</td>
</tr>
<tr>
<td>Issue a /STA db ACCESS=RO after the reorg.</td>
<td>EOJACCES(RO)</td>
</tr>
<tr>
<td>Start the database in the status it was when the reorg started.</td>
<td>EOJACCES()</td>
</tr>
</tbody>
</table>

Internal default: Start the database in the access it was before the reorg.

The EOJACCES keyword enables you to specify what access you wish the database to be started in online, when the REORG job finishes. If you want the access to be the same as it was when the REORG job started, do not specify this keyword. (This is the normal mode of operation.) If you want the database to be brought back online after the REORG in a specific access status, regardless of the access status it was when the REORG started, use this keyword.
FBFF

The Free Block Frequency Factor (FBFF) keyword is valid with the RELOAD, USERLOAD, and REORG commands. You can specify this option in PLUSIN; it is not available as a global option. Table 131 describes the keyword.

Table 131  FBFF keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify an override value for the free block frequency factor specified in the DBD.</td>
<td>FBFF(n1,n2...n10)</td>
</tr>
</tbody>
</table>

The FBFF keyword may be used to override the free block frequency factor specified in the DBD.

You can specify a separate value for up to ten data set groups. The values in the list correspond to the data set groups in the database. The first value is for the first data set group, etc.

The values in each subparameter can be a null or a number.

- number
  
  Indicates that the free block frequency factor for the corresponding data set group is to be reset using the specified number. The number must be in the range of 0 through 100, except 1.

- null
  
  Indicates that the value specified in the DBD for that data set will be used.

FFTOHAL

The Full Function to HALDB (FFTOHAL) keyword is valid with the LOAD and UNLOAD commands. This option can only be specified in PLUSIN. Table 132 on page 435 describes the keyword.
The FFTOHAL(YES) keyword requires the use of DFSURGU1(LONG).

**NOTE**
To avoid open failures, database DSN must have DISP=SHR or use DYNALLOC(Y).

The Full Function to HALDB migrate process supports the following types of logical relationships:

- unidirectional with direct pointers
- unidirectional with symbolic pointers
- bidirectional logically paired with direct pointers
- bidirectional logically paired with symbolic pointers
- bidirectional virtually paired

**Unidirectional with direct pointers**

The FFTOHAL keyword is not required for unidirectional direct logical relationships. The logical child points to the logical parent in both Full Function and HALDB. The standard DFSURGU1(LONG) format unload file carries this pointer.

Dependent on the amount of data, the buffers might have to be tuned using the DFSVSAMP DD card. In order to specify additional buffers using the DFSVSAMP DD card, you must add the following special DD to the job step:

```c
//SKEEPVS DD DUMMY
```
Unidirectional with symbolic pointers

The FFTOHAL keyword is required for logical relationships with unidirectional symbolic pointers. HALDB supports direct pointers only. Therefore this conversion is supported by finding the RBA of the logical parent at unload and saving that value in the unload file (this is not done in a standard unload file). The logical parent acquires an ILK that is built with the old RBA and the logical child acquires its logical parent EPS pointer built with this same old RBA. Normal HALDB pointer healing completes the process.

Bidirectional logically paired with direct and symbolic pointers

The FFTOHAL keyword is required for logical relationships with bidirectional direct pointers. In a full function logical relationship, the logical child points to its logical parent. In HALDB, bidirectional logically paired logical children point directly to their paired logical child. This conversion is supported by finding the RBA of the paired logical child and storing it in the unload file (this is not done in a standard unload file). Both of the paired logical children acquire an ILK that is built from their old RBA, and both paired logical children get an EPS pointer built with the old RBA value from their paired segment. Normal HALDB pointer healing completes the process.

Bidirectional virtually paired segments

The FFTOHAL keyword is required for bidirectional virtually paired logical relationships. In a set of full function databases with virtually paired logical relationships, one of the paired logical children does not physically exist. In HALDB, a virtually paired logical relationship is converted to bidirectional logically paired relationship with direct pointers. The unload process retrieves the data for the virtual logical child by reading the paired segment, then builds a record in the unload file for the virtual segment.

WARNING

An unload file built with the FFTOHAL keyword is not usable to reload back into a Full Function database. An unload file built with the FFTOHAL keyword is not usable by IBM reload.
FILSZ

The File Size (FILSZ) keyword is valid with the Index Build (HIU) function. FILSZ suppresses the use of SORTWORK. This keyword is available in PLUSIN; it is not available as a global option. Table 133 describes this keyword.

Table 133  FILSZ keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build an index</td>
<td>index</td>
</tr>
<tr>
<td>Estimated number of records.</td>
<td>no</td>
</tr>
<tr>
<td>Limit: between 1 and 4,294,967,295</td>
<td></td>
</tr>
</tbody>
</table>

The following example shows the syntax for the FILSZ keyword:

```
FILSZ(INDEXDBD1,100,INDEXDBD2,200,......INDEXDBDXn,99999)
```

When you build indexes for a database, you can use the FILSZ keyword to specify an estimated number of records for each index file. The FILSZ keyword enables you to specify estimated record counts for as many as 32 indexes.

**WARNING**

If you enter an incorrect value for the FILSZ parameter, the Sort utility performance can be degraded. An incorrect value can also cause wasted DASD space or termination when work space is dynamically allocated. Therefore, it is important to update the record count value whenever the number of records to be sorted changes significantly.

For more information about sorting with the Index Build function, see “Sort considerations” on page 249.

FIXED

The FIXED keyword is valid with the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or PLUSIN. Table 134 on page 438 describes the keyword.
Specify FIXED(Y) to allow fixed-length sorting of logical relationship records. Fixed-length sorts will improve performance but might increase the sort work space required.

### FMTALL

The FMTALL keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), GROUP, and USERLOAD commands. You can specify this option in PLUSIN. Table 135 describes the FMTALL keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow fixed-length sorting for logical relationship records.</td>
<td>FIXED(Y)</td>
</tr>
<tr>
<td>Do not allow fixed-length sorting for logical relationship records.</td>
<td>FIXED(N)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> (N)</td>
<td></td>
</tr>
</tbody>
</table>

### Table 134 Fixed-length sorting keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format all data sets completely.</td>
<td>FMTALL(YES)</td>
</tr>
<tr>
<td>Format selected data sets completely.</td>
<td>FMTALL(mm,...) or FMTALL(YES, mm,...)</td>
</tr>
<tr>
<td>Use mm to identify the data set group numbers of the data sets that will be formatted completely.</td>
<td></td>
</tr>
<tr>
<td>Format all data sets through the last used block.</td>
<td>FMTALL(NO) or omit the FMTALL keyword</td>
</tr>
<tr>
<td><strong>Internal Default:</strong> NO</td>
<td></td>
</tr>
</tbody>
</table>

The Reload function lets you control if the unused space in selected database data sets is formatted. You can specify FMTALL for all data set groups in the database or for selected data set groups. When you specify FMTALL for a data set group, the data set will be completely formatted, regardless of the number of blocks used. Otherwise the data set will be formatted only through the last used block.

If FMTALL is specified for the first data set group of a HDAM or PHDAM database, FMTRAA(YES) is forced. For more information, see “FMTRAA” on page 439.

For partitioned databases, the FMTALL specification applies to all partitions.
The FMTRAA keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. You can specify this option in a Reload function global options module or in PLUSIN. Table 136 describes the keyword.

### Table 136 Format the root addressable area keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Format the entire HDAM or PHDAM RAA.</td>
<td>FMTRAA(YES)</td>
</tr>
<tr>
<td>Format the RAA only to the block used.</td>
<td>FMTRAA(NO)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> YES</td>
<td></td>
</tr>
</tbody>
</table>

The Reload function lets you control whether it formats the complete HDAM or PHDAM root addressable area (RAA). When you specify FMTRAA(YES), the complete RAA is formatted, regardless of the number of blocks used. When you specify FMTRAA(NO), the RAA is formatted through the last block used. If the overflow area is used, the value you specified is ignored and the complete RAA is always formatted.

You can set the default value for the FMTRAA option by using the Reload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

---

The FORCELP keyword is valid with REORG (Online Reorg function and Reorg function) command and the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or PLUSIN. Table 137 describes the keyword.

### Table 137 Force logical parent counter field update keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change counter fields where you are deleting the logical child segments</td>
<td>FORCELP(Y)</td>
</tr>
<tr>
<td>during the reorganization process.</td>
<td></td>
</tr>
<tr>
<td>Do not change counter fields where you are deleting the logical child</td>
<td>FORCELP(N)</td>
</tr>
<tr>
<td>segments during the reorganization process.</td>
<td></td>
</tr>
<tr>
<td><strong>Internal default:</strong> N</td>
<td></td>
</tr>
</tbody>
</table>
The Prefix Update function does not update any logical parent (LP) counter fields to zero. You must specify FORCELP(Y) during Prefix Resolution for the Prefix Update function to change counter fields where you are deleting logical child (LC) segment occurrences for an LP segment during the reorganization process. FORCELP(Y) is valid for a DBIL-type reorganization only.

If FORCELP(Y) is specified, you cannot run a single-step reorganization, a reorganization in which the Prefix Resolution/Update function is integrated into the Reorg function.

**NOTE**
Specifying FORCELP(Y) degrades the performance.

---

The FREQ keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 138 describes the keyword.

**Table 138  Image copy monitoring messages frequency keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify how often the Image Copy function writes a message to the processing log or system console about the number of records it has read or written.</td>
<td>FREQ(nnnnnnnn)</td>
</tr>
<tr>
<td></td>
<td>nnnnnnnn is 0–99999999</td>
</tr>
</tbody>
</table>

Internal default: FREQ(10000)

For example, if you specify FREQ(1000), the Image Copy function writes a message after writing each 1000 records (1000 records, 2000 records, 3000 records, and so forth) until all records have been processed. At the end of record processing, the Image Copy function writes a message giving the total number of records processed.

If you specify FREQ(0), no messages are written.

For more information, see “Monitoring and reporting options” on page 334.
The Free Space Percentage Factor (FSPF) keyword is valid with the RELOAD, USERLOAD, REORG, OREORG, and ANALYZE commands. You can specify this option in PLUSIN; it is not available as a global option. Table 139 describes the keyword.

**Table 139  FSPF keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify a percentage of free space to qualify a block as free.</td>
<td>FSPF(n1,n2,...n10)</td>
</tr>
<tr>
<td>Internal default: 95</td>
<td></td>
</tr>
</tbody>
</table>

The FSPF keyword specifies the percentage of free space to indicate that a block is free. You can specify a maximum of 10 values (a separate value for up to ten data set groups). The first value is for the first data set group, etc.

The values in each subparameter can be a null or a number.

- number
  - Indicates that the free space percentage factor for the corresponding data set group is to be reset using the specified number. The number must be in the range of 0 through 99.

- null
  - Indicates that the value specified in the DBD for that data set will be used.

The FSPF keyword may be used to override the free space percentage factor specified in the DBD.

Online/Defrag uses the free blocks as the work area to reorganize database records. You can lower this number if a reorganization fails or if a lesser value will find more free blocks. A lesser value might result in using more blocks to reorganize the record.
The FUNC keyword is optional on the GROUP command. This keyword is available in ICPSYSIN; it is not available in global and database-specific options modules. Table 140 describes the keyword.

**Table 140  Image copy group function keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify the Image Copy function to perform for all data set groups defined with the DBDSGRP keyword.</td>
<td>FUNC(IC)</td>
</tr>
<tr>
<td></td>
<td>FUNC(AIC)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

Use the FUNC keyword in conjunction with the DBDSGRP keywords to identify the function to be performed for all data set groups defined within the database data set group (as identified with the DBDSGRP keyword).

The FUNC keyword has no meaning and is ignored when specified with the NAME keyword.

**GSGNAME**

The GSGNAME keyword is valid with the BUILD, REORG, and COPY commands, and the Prefix Resolution/Update function. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 141 describes the keyword.

**Table 141  GSG name keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the specified name for the GSG used in RSR.</td>
<td>GSGNAME(name)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> Use the GSG name used in system generation.</td>
<td></td>
</tr>
</tbody>
</table>

The GSGNAME keyword specifies the name of global services group (GSG) used for remote site recovery (RSR).

This keyword is valid for IMS Version 5.1 and later.
HDAMFSPF

The HDAMFSPF keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. You can specify this option in a Reload function global options module or in PLUSIN. Table 142 describes the keyword.

Table 142  Free space percentage factor keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ignore the FSPF.</td>
<td>HDAMFSPF(IGNORE)</td>
</tr>
<tr>
<td>Honor the FSPF.</td>
<td>HDAMFSPF(NO)</td>
</tr>
<tr>
<td>Internal default:</td>
<td>NO</td>
</tr>
</tbody>
</table>

One of the DBD parameters for an HDAM or PHDAM database is the free space percentage factor (FSPF), which allows you to specify the minimum percentage of each block to leave free at reload time. This percentage applies to each block in the RAA and the overflow area. The Reload function allows you to specify whether to honor or ignore the FSPF. When you specify HDAMFSPF(IGNORE), the FSPF is ignored in the RAA but used in the overflow area at reload time.

You can set the default value for the HDAMFSPF option by using the Reload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

HDSORT

The HDSORT keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. This keyword is available in PLUSIN; it is not available as a global option. Table 143 describes the keyword.

Table 143  Invoke HD Sort keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke the standard HD Sort utility.</td>
<td>HDSORT(YES)</td>
</tr>
<tr>
<td>Do not invoke the standard HD Sort utility or the RAP Sequencer.</td>
<td>HDSORT(NO)</td>
</tr>
<tr>
<td>Allow the product to choose the optimum method (standard HD Sort utility or RAP Sequencer) if it can determine that a sort is needed.</td>
<td>HDSORT(AUTO)</td>
</tr>
<tr>
<td>Internal default: AUTO</td>
<td></td>
</tr>
</tbody>
</table>
You can include the HDSORT keyword with the RELOAD command or the REORG command, or you can invoke the HD Sort utility as a stand-alone utility before reloading the database. For more information, see “HD Sort utility” on page 227.

The HDSORT keyword invokes the standard HD Sort utility or the BMC RAP Sequencer, depending on which value you specify.

The HD Sort utility sorts an HD Unload data set into HDAM, PHDAM, HIDAM, PHIDAM, or HISAM order. It is a functional alternative to the IMS System Utilities/Database Tools HDAM Physical Sequence Sort/Reload (PSSR) utility.

The RAP Sequencer is an alternative to the HD Sort utility. The RAP Sequencer does not sort data; it places roots in root anchor point (RAP) sequence. Because the RAP Sequencer does not actually sort data, performance is better than the HD Sort utility. You cannot explicitly invoke the RAP Sequencer. When you specify HDSORT(AUTO), the product determines whether a sort is needed and which sort method is appropriate.

Specify HDSORT(AUTO), or accept it as the default, in the RELOAD or REORG job step if you did not invoke the HD Sort utility before reloading the database and you performed any of the following tasks:

- convert an HDAM database to HIDAM or HISAM (RELOAD only)
- convert an PHDAM database to PHIDAM (RELOAD only)
- convert a HIDAM or HISAM database to HDAM (RELOAD only)
- convert a PHIDAM database to PHDAM (RELOAD only)
- change the key of a root segment
- change the randomizer parameters for an HDAM or PHDAM database (except maximum insert bytes)

When you specify HDSORT(YES), the standard HD Sort utility is always invoked.

When you specify HDSORT(AUTO), the product determines whether the data needs to be sorted or resequenced and invokes the standard HD Sort utility or the RAP Sequencer, depending on what changed in the DBD. The product determines whether to sort the data or resequence the RAPs by comparing the new DBD to the DBD named in the OLDDBD keyword or the BMC unload header record DBD control blocks. If the product detects a DBD change that requires a sort and the following criteria are not met, the standard HD Sort utility is invoked. If the data is not in randomized sequence and the following criteria are met, the RAP Sequencer is invoked because it provides better performance:

- The old DBD and the new DBD are HDAM or PHDAM.
- The old DBD and the new DBD use the DFSHDC40 randomizer.
- The root key position or length did not change.
- The root key is not compressed.
The total number of RAPs increased.
- You did not specify USERHDR when creating the HD Unload file.
- The HD Unload file was created by the Unload function.

If the data was resequenced and the criteria for the RAP Sequencer was not met, the HD Sort utility must be invoked to sort the data before it is reloaded. Therefore, if any data in the key changed and the DBD did not change, or if any other changes occurred that the product cannot detect by examining the old DBD or the unload header record DBD control blocks, you must specify HDSORT(YES). Specifying HDSORT(AUTO) does not guarantee that a sort is performed.

**DBD changes**

If you make any changes that resequence the roots and you are using the Reload function, the Reorg function, or the Online Reorg function to reload the database, the records must be in randomized order or key sequence order.

---

**WARNING**

If you are changing the key by using a user exit, specify the user exit during the unload step rather than the reload step. HD Sort is always invoked before the user exit in the Reload function.

---

If you are reloading the resequenced data with the Reload function and you executed a stand-alone sort of the data before reloading it, you can specify HDSORT(NO) to avoid an unnecessary sort. (The default is HDSORT(AUTO)).

If you are using the Online Reorg function or the Reorg function to reorganize the database and you specified the OLDDBD keyword on the REORG command, the product attempts to sort or resequence the data. Specify HDSORT(YES) or HDSORT(AUTO) when you specify the OLDDBD keyword.

If you specify the DFSDBUX1 keyword and you are implementing or changing a DFSDBUX1 exit between the unload and the reload, the DFSDBUX1 exit will change the root key.

**HD Unload file types**

If you are using an HD Unload file that was not created by the Unload function, do not specify HDSORT(AUTO). Specify HDSORT(YES) if you need the data to be sorted and the unload file is a standard HD Unload file created by other unload utilities.

Table 144 on page 446 shows restrictions for the HDSORT keyword.
If the database uses the Custom Huffman compression technique of the BMC DATA PACKER®/IMS product and key compression, and if you are running the HD Sort utility as a stand-alone utility, the database data set that contains the root must be specified in the JCL.

### Reload function API

HDSORT(AUTO) has a slightly different meaning when used with the Reload function API USERLOAD command. If you specify HDSORT(AUTO) or accept it as the default, the following values are used:

- For HDAM and PHDAM databases, HDSORT(YES) is used.
- In all other cases, HDSORT(NO) is used.

### HIPO

The HIPO option is valid with the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or PLUSIN. Table 145 describes the option.

#### Table 145  High performance option

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the High Performance option.</td>
<td>HIPO(Y)</td>
</tr>
<tr>
<td>Do not use the High Performance option.</td>
<td>HIPO(N)</td>
</tr>
</tbody>
</table>

**NOTE**

There are no restrictions for USERHDR (SEGNAME | SEGCODE) when used with HDSORT(YES).

Table 144  HDSORT keyword restrictions

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDSORT(YES) or</td>
<td>You cannot process HD Unload files that were created with the following keywords or formats:</td>
</tr>
<tr>
<td>HDSORT(AUTO)</td>
<td>- DFSURGU1(FRMT1, FRMT2, or FRMT3)</td>
</tr>
<tr>
<td></td>
<td>- USERHDR</td>
</tr>
<tr>
<td></td>
<td>- HSSR FRMT *F1, *F2, or *F3 control statements</td>
</tr>
</tbody>
</table>
The High Performance option (HIPO) passes the output from Parent-Child Resolution (phase 1) directly to Logical Twin Resolution (phase 2) using virtual storage internal buffering. Better performance is obtained, but more virtual storage is required and the job cannot be restarted except at the beginning.

The subkeywords shown in Table 146 are valid with HIPO(Y):

### Table 146 HIPO subkeywords

<table>
<thead>
<tr>
<th>Subkeyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFSIZE</td>
<td>Because the High Performance option uses internal buffers to pass data between the two prefix resolution phases, the size and number of buffers can affect performance. A too-small buffer size can cause excessive passing of buffers between sorts. In testing, 4 KB (4096 bytes) blocks have been found effective. The default is 4K.</td>
</tr>
<tr>
<td>NBUF</td>
<td>The number of buffers available affects the performance of the High Performance option. Too few buffers cause excessive waiting for buffers, while too many affect sort performance. BMC recommends that you use 16 to 20 buffers. The default is 16.</td>
</tr>
<tr>
<td>SIZE</td>
<td>The High Performance option schedules two sorts concurrently, one for the Parent-Child Resolution phase and another for the Logical Twin Resolution phase. Limit the amount of virtual storage allocated to the first sort. The value specified depends upon the amount of virtual storage allocated in the region and the amount of real storage. BMC recommends that you use 2M. The default is 500,000 (not 500K).</td>
</tr>
<tr>
<td>SORTDD</td>
<td>Because the two sorts run concurrently for the HIPO, a four-character prefix for the second sort’s data sets must be provided. This four-character prefix replaces the “SORT” in the SORTWKnn data set ddnames, for example SRT2WK01. The default is SRT2.</td>
</tr>
</tbody>
</table>

Specify subkeywords as follows:

```
HIPO(Y) NBUF(24)
```

### HIST

The HIST keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 147 describes the keyword:

### Table 147 Image copy history generations to retain keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify a limit for the generations of historical information that the Image Copy function maintains in the PDX for each execution of the Image Copy function.</td>
<td>HIST(nnnn)</td>
</tr>
<tr>
<td>Internal default: HIST(9999)</td>
<td>nnnn can be 0–9999</td>
</tr>
</tbody>
</table>
Each generation requires 512 bytes in the PDX. As you maintain more generations, more space is required.

HIST(0) means no historical statistics are to be generated for this execution of the Image Copy function. Statistics from previously accumulated histories are retained in the PDX.

HIST(9999) means you want to keep up to 9,999 generations.

For more information, see “Monitoring and reporting options” on page 334.

**HSAMOUT**

The HSAMOUT option is valid with the UNLOAD command. You can specify this option in an Unload function global options module or PLUSIN. Table 148 describes the HSAMOUT options.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an HSAM database (provide the DBD name).</td>
<td>HSAMOUT(dbdName)</td>
</tr>
<tr>
<td>Do not create an HSAM database.</td>
<td>HSAMOUT() (or do not code keyword)</td>
</tr>
</tbody>
</table>

**Internal default:** Blank (do not create an HSAM database).

The HSAMOUT option allows you to create an HSAM database from the database that is being unloaded. To use the HSAMOUT option, specify the DBD name of the HSAM DBD. You must also provide a DD statement for the HSAM database in your JCL. The ddname must match the name specified in the DD2 operand of the DATASET macro within the DBD.

It is not necessary for the segment definitions in the DBD being unloaded to match those in the HSAM DBD (for example, all segment names appearing in one DBD do not have to appear in the other DBD). The parent/child relationships must be the same. If the DBD being unloaded contains segment names that are not included in the HSAM DBD, those segments are ignored.

If the segment length of the segment being unloaded is different from the segment length definition in the HSAM DBD, the segment placed into the HSAM database is truncated or padded with binary zeros.
IC

The IC keyword is optional on the GROUP command. It is required on the IC and AIC command if it is not coded on the GROUP command. This keyword is available in ICPSYSIN; it is not available as a global option. Table 149 describes the keyword.

Table 149  Output image copy ddname keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the ddname of each output image copy data set to write.</td>
<td>IC(ddname,ddname,...)</td>
</tr>
<tr>
<td>Dynamically allocate each output image copy data set</td>
<td>IC(<em>,</em>,...)</td>
</tr>
<tr>
<td>Internal default: none</td>
<td></td>
</tr>
</tbody>
</table>

Use the IC keyword to identify the output image copy data set or data sets. Code the ddname if the output image copy data set is allocated in the JCL. Code an asterisk (*) to dynamically allocate the output image copy data set; you must also code ICALLOC(Y) in the ICPSYSIN data set (as described in “ICALLOC” on page 450). One copy of the output image copy data set is written to each ddname listed. On a GROUP command, the output image copy data sets must be dynamically allocated (you cannot code specific ddnames).

You can code a maximum of 10 ddnames or asterisks. Separate the ddnames or asterisks with commas.

For more information, see “Dynamic allocation of image copy data sets” on page 326.

ICACC

The ICACC keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 150 describes the keyword.

Table 150  Output image copy access method keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use SAM processing to write output image copy data sets.</td>
<td>ICACC(1)</td>
</tr>
<tr>
<td>Use EXCP processing to write output image copy data sets.</td>
<td>ICACC(2)</td>
</tr>
<tr>
<td>Internal default: ICACC(2)</td>
<td></td>
</tr>
</tbody>
</table>

Use the ICACC keyword to select the access method to use for writing output image copy data sets.
You can use EXCP processing (ICACC(2)) to write data sets to tape.

IMAGE COPY PLUS always uses SAM processing (ICACC(1)) to write data sets residing on DASD.

You can use the ICBUF keyword to control buffer usage, as described in “ICBUF” on page 451.

For more information, see “Output image copy write options” on page 331.

**ICALLOC**

The ICALLOC keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 151 describes the keyword.

### Table 151  Dynamically allocate output image copy keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamically allocate the output image copy data sets if a DD statement is not included in the JCL. Dynamically deallocate the output data set when processing is complete.</td>
<td>ICALLOC(Y)</td>
</tr>
<tr>
<td>Use JCL to allocate the output image copy data sets. The data sets are deallocated when the job step terminates.</td>
<td>ICALLOC(N)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: ICALLOC(N)</td>
<td></td>
</tr>
</tbody>
</table>

ICALLOC(N) is ignored when specified with IC(*), which forces dynamic allocation of the output image copy data set. ICALLOC(N) specified with IC(ddname) causes an abend if you do not include a DD statement in the JCL. For more information, see “IC” on page 449.

For more information about dynamic allocation, see “Dynamic allocation of image copy data sets” on page 326.
ICAVGREC

The ICAVGREC keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 152 describes the keyword.

### Table 152 Output image copy SMS AVGREC keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocate DASD space in blocks when dynamically allocating new Storage Management Subsystem (SMS) output image copy data sets.</td>
<td>ICAVGREC( ) (blank value)</td>
</tr>
<tr>
<td>Allocate DASD space in logical records.</td>
<td>ICAVGREC(U)</td>
</tr>
<tr>
<td>Allocate DASD space in logical records. The primary and secondary values specified with the ICSPACE keyword specify space in multiples of 1,000 logical records.</td>
<td>ICAVGREC(K)</td>
</tr>
<tr>
<td>Allocate DASD space in logical records. The primary and secondary values specified with the ICSPACE keyword specify space in multiples of 1,000,000 logical records.</td>
<td>ICAVGREC(M)</td>
</tr>
</tbody>
</table>

Internal default: ICAVGREC( )

The Image Copy function ignores this keyword under the following conditions:

- the data set to be allocated already exists
- SMS is inactive
- you do not request dynamic allocation (ICALLOC(N))
- you specify the first subparameter of ICSPACE as CYL or TRK

For more information, see “Dynamic allocation of image copy data sets” on page 326. See also the AVGREC parameter in the IBM MVS/ESA JCL Reference manual.

ICBUF

The ICBUF keyword is optional on the GLBL and GROUP commands. It is required on the IC and AIC commands if it is not specified on the GLBL or GROUP command. This keyword is available in ICPSYSIN; it is not available as a global option. Table 153 describes the keyword.

### Table 153 Output image copy buffer algorithm keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the output image copy buffer option.</td>
<td>ICBUF(value)</td>
</tr>
</tbody>
</table>

Internal default: ICBUF(4)
Specify one of the following values to control the number of buffers that the Image Copy function allocates for each output image copy data set:

Table 154  ICBUF values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Tape or Disk: Use the MVS default or, if specified, the number of buffers on the DCB=BUFNO= parameter in the JCL input.</td>
</tr>
<tr>
<td>1</td>
<td>Tape: Use four buffers for SAM or EXCP. Disk: Use the number of buffers equal to one track of DASD space plus one buffer.</td>
</tr>
<tr>
<td>2</td>
<td>Tape: Use five buffers for SAM or eight buffers for EXCP. Disk: Use the number of buffers equal to two tracks of DASD space plus one buffer.</td>
</tr>
<tr>
<td>3</td>
<td>Tape: Use 9 buffers for SAM or 16 buffers for EXCP. Disk: Use the number of buffers equal to four tracks of DASD space plus one buffer.</td>
</tr>
<tr>
<td>4</td>
<td>Tape: Use 17 buffers for SAM or 32 buffers for EXCP. Disk: Use the number of buffers equal to one cylinder of DASD space plus one buffer.</td>
</tr>
</tbody>
</table>

The Image Copy function allocates a minimum of 4 and a maximum of 255 buffers.

Generally, performance increases when you use more buffers. However, if system paging is heavy, the reverse might be true.

All buffers reside above the 16 MB line, whether you specify SAM or EXCP processing with the ICACC keyword (as described in “ICACC” on page 449). SAM processing allocates a set of buffers for every output image copy data set you request. EXCP processing allocates one set of buffers for all output image copy data sets, regardless of the number of image copies you request.

For more information, see “Output image copy write options” on page 331.

The ICCATF keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 155 on page 453 describes the keyword.
The ICCATF keyword is ignored when you specify ICGDG(Y). In this case, the output image copy data set is always cataloged.

**NOTE**

BMC recommends that you use ICCATF(Y) if DBRC is used and has been initialized with the CATDS option, which specifies that DBRC expects recovery data sets to be cataloged.

For more information, see “Dynamic allocation of image copy data sets” on page 326.

The ICCATF keyword is ignored when you specify ICGDG(Y). In this case, the output image copy data set is always cataloged.

**Table 155  Catalog output image copies keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create catalog entries for dynamically allocated output image copies.</td>
<td>ICCATF(Y)</td>
</tr>
<tr>
<td>Do not create catalog entries for dynamically allocated output image copies.</td>
<td>ICCATF(N)</td>
</tr>
<tr>
<td><strong>Internal default: ICCATF(Y)</strong></td>
<td></td>
</tr>
</tbody>
</table>

**ICDATACL**

The ICDATACL keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 156 describes the keyword.

**Table 156  Output image copy SMS DATACLAS keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the DATACLAS (data class) when you dynamically allocate new SMS image copy output data sets.</td>
<td>ICDATACL(dataclas)</td>
</tr>
<tr>
<td>Allocate the data set without a DATACLAS specification.</td>
<td>ICDATACL(dataclas)</td>
</tr>
<tr>
<td><strong>Internal default: ICDATACL( )</strong></td>
<td></td>
</tr>
</tbody>
</table>

Your storage administrator defines the allowed DATACLAS name.

This keyword is ignored when any of the following conditions exists:

- the data set to be allocated already exists
- SMS is inactive
- you do not request dynamic allocation (ICALLOC(N))
For more information, see “Dynamic allocation of image copy data sets” on page 326. See also the DATACLAS parameter in the IBM MVS/ESA JCL Reference manual.

### ICDSCB

The ICDSCB keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 157 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the name of the data set control block (DSCB) to use as a model for the generation data group (GDG) when you dynamically allocate the output image copy data sets.</td>
<td>ICDSCB(dscbName)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

A model data set control block is required when you create an image copy data set as a generation data group (as specified with the ICGDG keyword). For more information, see “ICGDG” on page 455.

For more information about dynamic allocation, see “Dynamic allocation of image copy data sets” on page 326.

### ICEXPDT

The ICEXPDT keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 158 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify an expiration date for output image copy data sets when they are dynamically allocated.</td>
<td>ICEXPDT(yyddd)</td>
</tr>
<tr>
<td></td>
<td>ICEXPDT/yyyy/ddd</td>
</tr>
<tr>
<td></td>
<td>yyyy is the 2-digit year</td>
</tr>
<tr>
<td></td>
<td>yyyy is the 4-digit year</td>
</tr>
<tr>
<td></td>
<td>ddd is the day of the year</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

Where:
- yyyy: 4-digit year
- yyyy: 2-digit year
- ddd: Day of the year
If you use ICRETPD, do not use ICEXPDT. For a description of the ICRETPD keyword, see “ICRETPD” on page 460.

**WARNING**

When you use dynamic allocation and you do not specify ICRETPD or ICEXPDT, your installation’s standards for expiration date apply.

For more information, see “Dynamic allocation of image copy data sets” on page 326.

---

**ICGDG**

The ICGDG keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 159 describes the keyword.

**Table 159  Create output image copy as GDG keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create image copy data sets as GDGs when the output image copy data sets are dynamically allocated.</td>
<td>ICGDG(Y)</td>
</tr>
<tr>
<td>Do not create image copy data sets as GDGs.</td>
<td>ICGDG(Y)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> ICGDG(N)</td>
<td></td>
</tr>
</tbody>
</table>

When you specify ICGDG(Y), IMAGE COPY PLUS dynamically allocates the image copy data sets as generation +1 and concatenates the GDG number to the end of the data set name as follows:

```
'prefix. ICn.DBD.DSB.Gnnnn.Vnn'
```

Also, if you specify ICGDG(Y), you must provide a model data set control block with the ICDSCB keyword. For more information, see “ICDSCB” on page 454.

For more information, see “Dynamic allocation of image copy data sets” on page 326.

---

**ICLIKE**

The ICLIKE keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 160 on page 456 describes the keyword.
This keyword is ignored when any of the following conditions exists:

- the data set to be allocated already exists
- SMS is inactive
- you do not request dynamic allocation (ICALLOC(N))

For more information, see “Dynamic allocation of image copy data sets” on page 326. See also the LIKE parameter in the IBM MVS/ESA JCL Reference manual.

### ICMGMTCL

The ICMGMTCL keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 161 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the name of the MGMTCLAS (management class) to use when you</td>
<td>ICMGMTCL(className)</td>
</tr>
<tr>
<td>dynamically allocate new SMS image copy output data sets.</td>
<td></td>
</tr>
<tr>
<td>Allocate the data set without a MGMTCLAS specification.</td>
<td>ICMGMTCL( )</td>
</tr>
<tr>
<td><strong>Internal default</strong>: ICMGMTCL( )</td>
<td></td>
</tr>
</tbody>
</table>

Your storage administrator defines the allowed MGMTCLAS names.

This keyword is ignored when any of the following conditions exists:

- the data set to be allocated already exists
- SMS is inactive
- you do not request dynamic allocation (ICALLOC(N))

For more information, see “Dynamic allocation of image copy data sets” on page 326. See also the MGMTCLAS parameter in the IBM MVS/ESA JCL Reference manual.
The ICNEEDED keyword is valid with the BUILD, REORG (Online Reorg function and Reorg function), and GROUP commands. You can specify this option in an Index Build function or Reorg function global options module or in PLUSIN. Table 162 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn the ICNEEDED flag on for all secondary and primary index DBDs. (Index Build function)</td>
<td>ICNEEDED(ON)</td>
</tr>
<tr>
<td>Turn the ICNEEDED flag off for all secondary and primary index DBDs. (Index Build function)</td>
<td>ICNEEDED(OFF)</td>
</tr>
<tr>
<td>Turn the ICNEEDED flag on for all primary database, secondary index, and primary index DBDs. (Reorg function)</td>
<td>ICNEEDED(ON)</td>
</tr>
<tr>
<td>Turn the ICNEEDED flag off for primary database, secondary index, and primary index DBDs that are defined NONRECOV. (Online Reorg function and Reorg function)</td>
<td>ICNEEDED(OFF)</td>
</tr>
</tbody>
</table>

**Internal default**: Set according to RECOV and NONRECOV settings. If the database is set as RECOV, the default is ICNEEDED(ON). If the database is set as NONRECOV, the default is ICNEEDED(OFF).

The NOTIFY.REORG command turns the ICNEEDED flag on. The ICNEEDED keyword specifies whether the RECON data sets should have the Image Copy Needed (ICNEEDED) flag left on or turned off for the reorganized databases or the index DBD being loaded. If you do not specify this keyword, the default is set according to the DBRC RECOV and NONRECOV settings:

- If a database is set as RECOV, the default is ICNEEDED(ON).
- If a database is set as NONRECOV, the default is ICNEEDED(OFF).

For information about NONRECOV databases, see “DBRC and NONRECOV” on page 643.

When you specify ICNEEDED(OFF), a NOTIFY.UIC is issued for RECOV databases. For more information, see “NOTIFY.UIC and recovery” on page 642.

DBRC must be active for the ICNEEDED keyword to work.

You can set the default Index Build function value for the ICNEEDED option by using the Index Build function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library. You can set the default Reorg function value for the ICNEEDED option by changing the FRFSGLBL global options module. Sample JCL is in member #FRFGLBL of the sample library.
If a database is registered as NONRECOV, DBRC activity is also affected by the value you specified for the NONRECOV keyword in the DLIGSET0 module.

**Reorg function**

If you want the ICNEEDED flag to remain turned on after the NOTIFY.REORG for all reorganized databases (including indexes) after the reorganization completes, specify ICNEEDED(ON). For more information, see “Reorg function checking” on page 645.

**Index Build function**

The value of the ICNEEDED keyword determines the action the Index Build function takes with the ICNEEDED flag (based on RECOV/NONRECOV status), as shown in Table 163.

<table>
<thead>
<tr>
<th>When you specify</th>
<th>The ICNEEDED flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICNEEDED(ON)</td>
<td>The ICNEEDED flag that was set with NOTIFY.REORG is left on.</td>
</tr>
<tr>
<td>ICNEEDED(OFF)</td>
<td>The ICNEEDED flag is turned off when the NOTIFY.UIC is issued.</td>
</tr>
</tbody>
</table>

**ICP**

The ICP keyword is valid with the REORG command. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 164 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke the Image Copy function.</td>
<td>ICP(YES)</td>
</tr>
<tr>
<td>Do not invoke the Image Copy function.</td>
<td>ICP(NO)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> NO</td>
<td></td>
</tr>
</tbody>
</table>

The ICP keyword invokes the Image Copy function to take an image copy of the database as it is being reloaded. For complete information about the Image Copy function, see Chapter 11, “Image Copy function.”

When you specify ICP(YES), the Image Copy function is invoked to take the image copy. You can define image copy keywords in the ICPSYSIN DD statement.
You cannot create compressed image copies by using the Image Copy function. You cannot stack image copies for a database that has multiple data set groups or indexes.

**Online Reorg function**

The default value of ICP(YES) is required for Online Reorg function. You do not need to specify the ICP keyword in an Online Reorg function job. If you specify ICP(NO), Online Reorg function ignores the specification, forces the value to ICP(YES), and issues message BMC90454.

You may supply an ICPSYSIN DD statement with an IC or AIC command for each recoverable database on the ICPSYSIN DD statement, or the Online Reorg function will automatically create an IMAGE COPY for all databases being reorganized that are registered as RECOV in DBRC.

The Online Reorg function performs the image copy automatically during the Reload function or the prefix resolution and update execution, depending on the presence of logical relationships with direct pointers.

**Reorg function**

If ICP(YES) is specified, the Reorg function automatically performs the image copy.

No image copy is taken if prefix resolution and update is deferred. You must add the image copy step after the prefix resolution and update steps in your JCL.

If you specify SWAP(x,IC), the primary database and primary index are always treated as RECOV.

**ICPREF**

The ICPREF keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 165 on page 460 describes the keyword.
A data set name prefix is required when you dynamically allocate an output image copy data set by using the ICALLOC(Y) and IC(*) keywords (as described in “ICALLOC” on page 450 and “IC” on page 449). The prefix can be 1–33 characters. This prefix is the high-level qualifier of the data set name. The prefix can include periods and must conform to normal data set naming standards; it cannot end with a period.

For dynamic allocation, you must provide a data set name prefix or use a data set name user exit (ICP$DSN or RCU$DSN) that does not require a data set name prefix. For more information, see “Using the image copy Data Set Name user exit” on page 678.

Do not specify prefixes SYS, PLUS, and DFS when using preallocated GDGs; the Image Copy function ignores ddnames with these prefixes.

**WARNING**

Even though the ICPREF keyword allows up to 33 characters for identifying the data set name prefix, the entire data set name can be only 44 characters. Because of the concatenations, a GDG or stacked data set name could easily exceed the 44-character limit. A data set name with more than 44 characters causes the Image Copy function to abend.

For more information about dynamic allocation, see “Dynamic allocation of image copy data sets” on page 326.

**ICRETPD**

The ICRETPD keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 166 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify a retention period (in days) for output image copy data sets that are dynamically allocated.</td>
<td>ICRETPD(value)</td>
</tr>
</tbody>
</table>

*value is 0–9999*

**Table 166**  Output image copy retention period keyword

**Table 165**  Output image copy data set name prefix keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify a data set name prefix for the output image copy data set.</td>
<td>ICPREF(dsnprefix)</td>
</tr>
</tbody>
</table>

*Internal default: none*
You can specify either ICRETPD or ICEXPDT, but not both.

**WARNING**

If you use ICRETPD(0) with dynamic allocation, you could lose your output image copy. In most installations, ICRETPD(0) indicates that the output tape is a work tape that can be overwritten immediately. Your installation's standards for ICRETPD(0) might differ.

When you use dynamic allocation and you do not specify the ICRETPD or ICEXPDT keyword, your installation’s standards for retention period apply.

For a description of the ICEXPDT keyword, see “ICEXPDT” on page 454. For more information about dynamic allocation, see “Dynamic allocation of image copy data sets” on page 326.

### ICSFACE

The ICSFACE keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICSYSIN; it is also available in global and database-specific options modules. Table 167 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICSYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the amount of DASD space to allocate for a new non-SMS data set or to override the space allocation defined in the DATACLAS (data class) for an SMS data set.</td>
<td>ICSFACE(units,pri,sec)</td>
</tr>
<tr>
<td></td>
<td>ICSFACE(units,pri)</td>
</tr>
<tr>
<td>Use DATACLAS defaults for SMS data sets; use MVS defaults for non-SMS data sets.</td>
<td>ICSFACE( )</td>
</tr>
<tr>
<td><strong>Internal default:</strong> ICSFACE( )</td>
<td></td>
</tr>
</tbody>
</table>

Set the values for the ICSFACE keyword as shown in Table 168 on page 462.
Table 168  ICSPACE values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unit</td>
<td>Specify the unit to use for allocating the output space. You can set the value to cylinders (C or CYL), tracks (T or TRK), or a number (1–65535). If you specify a number, its meaning is defined by the ICAVGREC keyword. If you specify U, K, or M and SMS is active, unit is interpreted as the average record length. Otherwise, unit is interpreted as the average block size.</td>
</tr>
<tr>
<td>pri</td>
<td>Specify the number (1–16777215) of primary units to allocate.</td>
</tr>
<tr>
<td>sec</td>
<td>Specify the number (0–16777215) of secondary units to allocate.</td>
</tr>
<tr>
<td>blank</td>
<td>Use DATACLAS defaults for SMS data sets; use MVS defaults for non-SMS data sets.</td>
</tr>
</tbody>
</table>

For more information, see “Dynamic allocation of image copy data sets” on page 326. See also the SPACE parameter in the IBM MVS/ESA JCL Reference manual.

ICSTORCL

The ICSTORCL keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 169 describes the keyword.

Table 169  Output image copy SMS STORCLAS keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the name of a STORCLAS (storage class) to use when you dynamically allocate new SMS image copy output data sets.</td>
<td>ICSTORCL(className)</td>
</tr>
<tr>
<td>Allocate the data set without a STORCLAS specification.</td>
<td>ICSTORCL( )</td>
</tr>
<tr>
<td><strong>Internal default:</strong> ICSTORCL( )</td>
<td></td>
</tr>
</tbody>
</table>

Your system administrator defines the allowed STORCLAS names.

This keyword is ignored when any of the following conditions exists:

- the data set to be allocated already exists
- SMS is inactive
- you do not request dynamic allocation (ICALLOC(N))

For more information, see “Dynamic allocation of image copy data sets” on page 326. See also the STORCLAS parameter in the IBM MVS/ESA JCL Reference manual.
ICUNIT

The ICUNIT keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 170 describes the keyword.

Table 170  Output image copy unit keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the device unit name to use when dynamically allocating each output image copy data sets.</td>
<td>ICUNIT(unitName)</td>
</tr>
<tr>
<td>Specify the device unit name to use when dynamically allocating each output image copy data sets. Also specify the number of physical units to allocate.</td>
<td>ICUNIT(unitName,count)</td>
</tr>
</tbody>
</table>

Internal default: ICUNIT(TAPE,1)

For the unitName value, specify the device unit name. For the count value, specify the number of devices to allocate (1–59). For a tape device type, the Image Copy function automatically limits the count to 2.

For more information, see “Dynamic allocation of image copy data sets” on page 326.

ICVOLCNT

The ICVOLCNT keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 171 describes the keyword.

Table 171  Output image copy volume count keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Override your installation’s MVS specification on the maximum number of volumes allowed for an output image copy data set when they are dynamically allocated.</td>
<td>ICVOLCNT(value)</td>
</tr>
</tbody>
</table>

value is 0–255

Internal default: ICVOLCNT(0)

This option works like the volume-count subparameter that can be specified in JCL. You can specify any value from 0 through 255, but the actual value is 0, 5, 20 or 255, whichever is greater than or equal to the value you specify.

The ICVOLCNT keyword is ignored if you use it in conjunction with a volume serial list (specified with the ICVOLSER keyword). For more information, see “ICVOLSER” on page 464.
For more information about dynamic allocation, see “Dynamic allocation of image copy data sets” on page 326.

ICVOLSER

The ICVOLSER keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 172 describes the keyword.

Table 172 Output image copy volume serial keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify a list of one or more volume serial identifiers to use when you dynamically allocate new image copy output data sets.</td>
<td>ICVOLSER(volser,volser,...)</td>
</tr>
<tr>
<td>Internal default: ICVOLSER()</td>
<td></td>
</tr>
</tbody>
</table>

If you use the ICVOLSER keyword, the ICVOLCNT keyword has no meaning and is ignored. ICVOLSER has no meaning if the data set already exists. For more information about the ICVOLCNT keyword, see “ICVOLCNT” on page 463.

The value of the ICVOLSER keyword is the volume serial identifier. You can specify a string of up to 64 characters; separate each volume serial identifier with a comma.

For more information about dynamic allocation, see “Dynamic allocation of image copy data sets” on page 326.

IDCAMS

The IDCAMS keyword is valid with the BUILD, RELOAD, REORG, GROUP, and COPY commands. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 173 describes the keyword.

Table 173 IDCAMS keyword (part 1 of 2)

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify input file for IDCAMS VSAM processing.</td>
<td>IDCAMS(input)</td>
</tr>
<tr>
<td>Specify input and output files for IDCAMS processing.</td>
<td>IDCAMS(input,output)</td>
</tr>
<tr>
<td>Use the data set referenced by the AMSPDS DD statement as input for IDCAMS processing.</td>
<td>IDCAMS(*)</td>
</tr>
<tr>
<td></td>
<td>IDCAMS(*,output)</td>
</tr>
<tr>
<td>Reclaim DASD space after a Online Reorg function reorganization completes.</td>
<td>IDCAMS(input,output,Y)</td>
</tr>
</tbody>
</table>
The IDCAMS keyword allows you to perform IDCAMS DELETEs and DEFINEs for VSAM data sets in a job step instead of a previous job step.

**NOTE**
When the IDCAMS keyword is used with Reorg and Copy functions the DELETE/DEFINE statements should delete and define the shadow data sets.

You can specify the IDCAMS keyword as follows:

- IDCAMS(input) specifies the ddname of the input data set. The input file is a sequential file (or a member of a PDS) that contains IDCAMS statements.

- IDCAMS(input,output) specifies the ddnames of the input data set as well as the output data set for the IDCAMS utility. The output file is usually a SYSOUT data set. If the output parameter is omitted, the output is sent to the BMCMSG data set.

- When you specify IDCAMS(*) or IDCAMS(*,output), also include an AMSPDS DD statement that references the PDS that contains the IDCAMS DELETE/DEFINE statements. Specify the data set name of the PDS only; do not specify the member name. The Index Build function selects the members that match the index DBD ddnames. The Reorg function selects the members that match the main database, primary index, and secondary index DBD ddnames. The Reload function selects the members that match the main database and primary index DBD ddnames.

  The ddnames you specify with the IDCAMS keyword must be in the JCL.

**NOTE**
The input ddname cannot be SYSIN. The output ddname cannot be SYSPRINT or SYSOUT.

If you specify the IDCAMS keyword, you must specify an input file. The output file is optional.

If IDCAMS issues a return code other than 0 or 8, the BMC reorganization job terminates. Return code 8 indicates an attempt to delete a non-existent file.
**NOTE**

If you are not dynamically allocating the target/shadow database data sets (by using the DYNALLOC keyword), the data sets must be included in the JCL.

---

**Online Reorg function**

This section describes how to use the IDCAMS keyword in the Online Reorg function.

**Multiple database reorganizations (GROUP command)**

When using the IDCAMS keyword with the GROUP subcommand, DYNALLOC(...,y,.) is required if IDCAMS(...,....,Y) is specified.

**VSAM Data Sets**

If you preallocate all VSAM components for the shadow data sets before the Online Reorg function step, the IDCAMS keyword is not required. If you prefer to have the Online Reorg function invoke IDCAMS to DELETE/DEFINE the VSAM shadow data sets, specify the IDCAMS keyword.

**Reclaim Space**

You can specify an option to reclaim DASD space. The DASD space reclaimed is the space saved by reorganizing the database. For example, if the disorganized database has six extents but the reorganized database takes only two extents, the IDCAMS(input,output,Y) option causes the reorganized database to take only the two extents it needs, not the six extents it originally took.

During a Online Reorg function reorganization, the disorganized database is copied to the shadow database and then is unloaded and written to a temporary file. When the unload is complete, the Reload function reads the temporary file and reloads the records directly to the shadow database.

If you specify IDCAMS(input,output,N), the VSAM REUSE option is dynamically changed for the shadow database. All extent information for the shadow data set is preserved; the reorganized database will have the same extent information as the disorganized database. The VSAM REUSE option is changed back after the reorganization completes.

If you specify IDCAMS(input,output,Y), the reorganization job uses the IDCAMS input after the Copy phase instead of changing the VSAM REUSE option in the shadow database. IDCAMS input (if present) will always be used at the beginning of the reorganization to clear the shadow databases. This option redefines the reorganized data set with the new extent information.
OSAM Data Sets

All OSAM data sets, including shadow data sets, must be preallocated. The IDCAMS keyword is valid for both VSAM and OSAM data sets.

Reorg function

If you specify the IDCAMS keyword on the REORG command with SHARE(YES,READ), the Reorg function uses the data set specified by the IDCAMS keyword once to delete and define the VSAM shadow data sets at initialization if shadow data sets are used for data set name swapping.

MODEL is a parameter for the IDCAMS of the REORG command. The MODEL parameter is used to specify an existing cluster from which the attributes used to define the new cluster should be copied. BMC only supports SMS managed data sets since we do not allow VOLUME to be overridden.

MODEL can be used to create shadow secondary indexes with a two character partition number appended to the letter defined in the DYNALLOC keyword. When a subset of partitions is reorganized and no secondary index shadows are found that conform to the normal DYNALLOC naming convention, the online reorganization will automatically recognize that these special shadows are required. This special shadow naming convention allows multiple concurrent online reorganizations of subsets of the same HALDB database to run.

WARNING

You might encounter space problems if you use this keyword with non-SMS data sets.

If you specify the IDCAMS(MODEL) keyword and do not specify an input file, then you must set DYNALLOC(YES) on.

WARNING

If IDCAMS issues any non-zero return code, the BMC reorganization job terminates. Check for messages issued by IDCAMS.

If you have a ddname MODEL present in the job, the MODEL will be treated as an input data set. The following example shows the use of the MODEL parameter in the IDCAMS statement:

```
DEFINE CLUSTER ( -
NAME(A.B.C.Z) -
MODEL(A.B.C) -
)
```
**Index Build function**

If you specify the IDCAMS keyword on the BUILD command with DYNALLOC(YES,YES) to dynamically allocate the target data sets, the Index Build function uses the IDCAMS input data set to delete and define the indexes that are being built. If you are building the primary index and secondary indexes in the same step, you must supply DELETE and DEFINE statements for all data sets that you did not delete before the Index Build function job.

**Reload function**

OSAM data sets must be preallocated.

**ILDS**

The ILDS keyword is valid with the BUILD, RELOAD, REORG (FAST REORG FACILITY/EP), and GROUP commands. This keyword is used to specify the action taken on HALDB Indirect List Data Sets (ILDS). It is available in PLUSIN; it is not available as a global option. Table 174 and Table 175 on page 469 describe the keyword as it relates to each command.

**Table 174  ILDS keyword and the BUILD command**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the ILDS and secondary indexes.</td>
<td>ILDS(YES)</td>
</tr>
<tr>
<td>Build only the ILDS.</td>
<td>ILDS(ONLY)</td>
</tr>
<tr>
<td>Do not build the ILDS when building the secondary indexes.</td>
<td>ILDS(NO)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: NO</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE**

When ILDS(YES) is specified, the ILDS is not populated with records for segments which are secondary index targets. The following conditions apply to specifying ILDS(Yes) for the build command:

- If a database has no logical relationships, the ILDS will be empty and initialized.
- If there are logical relationships, the ILDS will contain only entries for the logically related segments.
Table 175  ILDS keyword and the RELOAD and REORG commands

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>■ An empty ILDS is created.</td>
<td>ILDS(INIT)</td>
</tr>
<tr>
<td>■ WF1 records are created for any segment that is a secondary index target segment. The WF1 records are used by the SECONDARY INDEX UTILITY to rebuild secondary indexes.</td>
<td></td>
</tr>
<tr>
<td>■ The reorganization number is set to 0001.</td>
<td></td>
</tr>
<tr>
<td>■ An ILDS is created containing records for any segment that is a secondary index target or that is involved in a logical relationship.</td>
<td>ILDS(BUILD)</td>
</tr>
<tr>
<td>■ No WF1 records are created.</td>
<td></td>
</tr>
<tr>
<td>■ The reorganization number is advanced.</td>
<td></td>
</tr>
<tr>
<td>Automatically determine if ILDS(INIT) or ILDS(BUILD) is the best option. For information about how the ILDS(AUTO) value is determined, see “Reload and Reorg functions” on page 469.</td>
<td>ILDS(AUTO)</td>
</tr>
<tr>
<td>Internal Default: AUTO</td>
<td></td>
</tr>
</tbody>
</table>

**Reload function**

If using the RELOAD command, the following rules apply to ILDS values:

■ The WF1 DD statement is not valid while using ILDS(BUILD).
■ USERLOAD is not valid with ILDS(BUILD).
■ A subset of partitions is not valid with ILDS(INIT).
■ Logically related databases cannot use ILDS(INIT).

If using the REORG command, the following rules apply to ILDS values:

■ The WF1 DD statement is ignored.
■ A subset of partitions is not valid with ILDS(INIT).
■ Logically related databases cannot use ILDS(INIT).

**Reload and Reorg functions**

For the RELOAD and REORG commands, the following rules apply for translating ILDS(AUTO):

■ If logical relationships are present, ILDS(AUTO) translates to ILDS(BUILD).
If secondary indexes are present and you are processing a subset of partitions, ILDS(AUTO) translates to ILDS(BUILD).

If logical relationships are not present, secondary indexes are present, and you are processing all partitions, ILDS(AUTO) translates to ILDS(INIT).

If logical relationships and secondary indexes are not present, ILDS(AUTO) translates to ILDS(INIT).

Index Build function

To build Indirect List Data Sets (ILDSs) for selected HALDB partitions, use the PART parameter to specify the partitions.

The following rules apply to building ILDS for selected HALDB partitions:

- Specify ILDS(ONLY).
- You cannot specify both the PART and SECINDEX parameter.
- The partition list must specify the list of primary database partitions.
- You cannot specify the following combinations:
  - ILDS(YES) and PRIMIND(ONLY)
  - ILDS(ONLY) and PRIMIND(YES)

The following JCL shows an example that results in ILDS for partitions CUST2A and CUST3A.

```jcl
BUILD DBD(CUSTOMER) ILDS(ONLY) PART(CUST2A,CUST3A)
```

Specifying ILDS(YES) builds the secondary indexes and the ILDS records for segments that are the target of a logical relationship only.

ILDS(ONLY) continues to build ILDS records for all segments that are targets of an EPS pointer.

IMSID

The IMSID keyword is valid with the UNLOAD, BUILD, DBSCAN, RELOAD, REORG (Online Reorg function and Reorg function), and COPY commands. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 176 on page 471 describes the keyword.
The IMSID keyword lets you specify the IMSID used to retrieve options from the PDX and store reports in the PDX. If you do not specify this keyword, the utility uses internal defaults as described in Table 176.

**NOTE**
If you are using the Online Reorg function and you want to override the IMSID specified on the EXEC statement or the RESLIB default, use the SSID keyword. The IMSID keyword is used only for PDX processing.

### Table 176  IMSID keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use “imsid” as the IMSID portion of the PDX key.</td>
<td>IMSID(imsID)</td>
</tr>
<tr>
<td>Internal default:</td>
<td></td>
</tr>
<tr>
<td>Unload function, Reload function, Reorg function, and Online Reorg function: Use the IMSID value specified in the EXEC statement’s PARM field. If the PARM field does not specify an IMSID, the utility uses the IMSID value from the IMS generation (DFSVC000 module)</td>
<td></td>
</tr>
<tr>
<td>Index Build function: Use the IMSID value from the IMS generation (DFSVC000 module)</td>
<td></td>
</tr>
</tbody>
</table>

The IMSID keyword lets you specify the IMSID used to retrieve options from the PDX and store reports in the PDX. If you do not specify this keyword, the utility uses internal defaults as described in Table 176.

### Table 177  Input data set keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>For the BUILD command, use the work file created by an HD reorganization reload utility as the input data set.</td>
<td>INDD(DFSURWF1)</td>
</tr>
<tr>
<td>For the COPY command, use to identify the ddname name of the database to be copied.</td>
<td>INDD(ddname)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

**BUILD command**

The INDD keyword specifies the ddname of the input data set that contains the secondary index records. If specified, the Index Build function uses this data set to load all secondary indexes; the primary database is not scanned. If you do not specify the INDD keyword, the Index Build function scans the database to gather the index records.
DFSURWF1 is the preferred value. DFSURIDX may also be specified, though it is not recommended. DFSURIDX is treated the same as a data set reference by //DFSURWF1. For example, if the file associated with the DFSURIDX DD statement has been presorted, the file will be sorted again.

If you specify DFSURWF1 DD DUMMY, the Index Build function creates empty index databases.

**COPY command**

The INDD keyword is used with the COPY command. The keyword specifies the ddname of the database to be copied. The value must match the ddname of the database in the corresponding DBD. If the database is a HALDB, the value must match the ddname of the partition to be copied; the value is verified in the RECONs.

You can use INDD to copy selected data set groups of a database or selected partitions of a HALDB. If you are reorganizing an entire database, BMC recommends dynamic allocation. For information about the DYNALLOC keyword, refer to page 430.

INDD cannot specify the ILDS of a HALDB; if DYNALLOC is used for HALDB, the ILDS is automatically copied.

**NOTE**

If you use INDD/OUTD, you cannot use dynamic allocation. BMC recommends dynamic allocation.

---

**INDDX**

The INDDX keyword is valid with the BUILD command. This keyword is available in PLUSIN; it is not available as a global option. Table 178 describes the keyword.

**Table 178 Read secondary index records from non-DFSURWF1 file**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read secondary index work file records from a data set other than DFSURWF1.</td>
<td>INDDX(ddname1(indexdb1), ddname2(indexdb2)...</td>
</tr>
<tr>
<td>Dynamically allocate secondary index work file created during a Reload function step, and read the records from that work file.</td>
<td>INDDX(AUTO)</td>
</tr>
<tr>
<td>Internal default: none</td>
<td></td>
</tr>
</tbody>
</table>
The INDDX keyword enables you to read secondary index records from a data set other than DFSURWF1. You can read from multiple data sets simultaneously. When the INDDX keyword is specified, secondary index records are not read from the DFSURWF1 file; the records for each secondary index are read from separate files.

If you specify the INDD keyword but do not include an INDDX keyword, all secondary index records are read from the DFSURWF1 data set. If you do not specify INDD or INDDX, the primary database is scanned.

The INDDX keyword does not support shared secondary indexes.

The Index Build function INDDX keyword is used with the data sets created by the Reload function OUTDDX keyword.

**INDDX(ddname1(…))**

Specify the input ddnames to read and the secondary index WF1 records to be read from them in the following format (maximum of 256 characters):

```
INDDX(ddname1(indexdb1), ddname2(indexdb2)…)
```

You must specify each secondary index that is being built in this job step, and each secondary index must have a different ddname.

**INDDX(AUTO)**

When you specify INDDX(AUTO), the DFSURWF1 file must have been created in the same job by a Reload function step that specified OUTDDX(AUTO). When you specify INDDX(AUTO), will dynamically allocate the files created in the Reload function step. All secondary indexes must be built in this job step.

**INPUT**

The INPUT keyword is valid with the REORG (online Reorg function and Reorg function) command. This keyword is available in PLUSIN; it is not available as a global option. Table 179 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execute a standard Reorg function reorganization.</td>
<td>INPUT(DB) or INPUT(DB,REORG)</td>
</tr>
<tr>
<td>Initialize an empty reorganized database (Online Reorg function).</td>
<td>INPUT(DB,INIT)</td>
</tr>
</tbody>
</table>
The INPUT keyword specifies the source of the input data for the reorganization. A Reorg function reorganization works as follows:

1. The Reorg function reads the unload data (from the disorganized database or from an HD Unload file, depending on the value of the INPUT keyword). If you specify, INPUT(DB,INIT), no unload data is read.

2. The Reorg function passes the unload records to the Reload function for reload. The Reload function reloads the database to shadow data sets or the original data sets, depending on the value of the INPUT keyword. If you specify, INPUT(DB,INIT), the shadow data sets are initialized.

3. While the Reload function is reloading the database, the Index Build function creates secondary indexes. If you specify, INPUT(DB,INIT), the secondary indexes are initialized.

4. If you specified ICP(YES), the Image Copy function is invoked to take an image copy of the database as it is being reloaded.

You can use the disorganized database as input, or you can use an HD Unload file.

**INPUT(DB) or INPUT(DB,REORG)**

When you specify INPUT(DB) or INPUT(DB,REORG), the Reorg function invokes the Unload function to unload the database. The original database is unloaded and reloaded to shadow data sets, or an image copy of the reorganized database is used to restore the original database. (Reloading to an image copy saves DASD space.) You can swap the data set names. This is a standard Reorg function reorganization.

**INPUT(DB,INIT)**

When you specify INPUT(DB,INIT), the original database is not read or copied. The Unload function passes an immediate end-of-file to the Reload function which initializes the shadow data sets to create an empty database. Changes to the original database that occur during the initialization are not captured. You can swap the data set names to use the initialized database online.
Chapter 14 User-controlled options

INPUT(UNLDFILE)

When you specify INPUT(UNLDFILE), you must create an HD Unload file in a separate step before the reorganization. The unload file must be in the IBM HD Unload format or the Unload function LONG, SHORT, or XSHORT format. HSSR formats 1, 2, and 3 and USERHDR formats are not accepted. The Reorg function reads the HD Unload file and reloads the database into the original database data sets, thus saving DASD space.

**NOTE**

This Reorg function option allows you to take advantage of concurrent image copies and/or the building of secondary indexes during the reload step. The Reload function does not support ICP(YES) and the building of secondary indexes.

The HD Unload file that you use as input must contain the entire database. You cannot use an HD Unload file that contains selected partitions of a database that uses the IMS/ESA Partition Support Product.

Table 180 shows keywords related to the INPUT(UNLDFILE) keyword.

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Effect on related keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXPAND</td>
<td>Not allowed. The job will abend</td>
</tr>
<tr>
<td>LPCK</td>
<td>Ignored</td>
</tr>
<tr>
<td>SWAP</td>
<td>Not allowed. The job will abend</td>
</tr>
<tr>
<td>UNLDD and UNLDDLST</td>
<td>Not allowed. The job will abend</td>
</tr>
<tr>
<td>SHARE(YES)</td>
<td>Not allowed. The job will abend</td>
</tr>
<tr>
<td>OLDDBD</td>
<td>Not allowed. The job will abend</td>
</tr>
<tr>
<td>PART</td>
<td>Not allowed. The job will abend</td>
</tr>
</tbody>
</table>

Because the reorganized database is reloaded into the original data sets, the original database will not be available in case of a failure. BMC recommends that you take an image copy of the disorganized database before beginning the reorganization.
When you specify INPUT(UNLDFILE), use reorganization JCL that is similar to the Reload function JCL (not the Reorg function JCL):

- Specify the data set names for the reorganized database (original data set names) using the DBD ddnames.

**NOTE**

This is the opposite of how a standard Reorg function reorganization works. Construct JCL that is similar to what you use for reorganization using stand-alone utilities.

- Specify a DFSUINPT DD statement (if the database is to be reloaded in non-parallel mode) or the appropriate number of DFSUINnn DD statements (if the database is to be reloaded in parallel mode) for each HD Unload file. You cannot specify DFSUINPT DD DUMMY.

If DBRC is active and the database data sets are registered, the output data set names must be registered to DBRC. If DBRC is active, the Reorg function completes all DBRC notifications.

**Empty database initialization with the Reorg function**

You can initialize an empty database by using the REORG command and making appropriate JCL statement modifications. To create a usable empty database, specify a DFSUINPT DD statement that points to an empty sequential data set. You cannot specify DFSUINPT DD DUMMY. Include the INPUT(UNLDFILE) keyword in the PLUSIN control statement data set, and follow all rules for using INPUT(UNLDFILE).

**Empty database initialization with the Online Reorg function**

To initialize an empty database with the Online Reorg function, use the INPUT(DB,INIT) option. This allows you to initialize an online database with minimum offline time. The shadow database is initialized so that the only time the database has to be offline is while the data set names are swapped and DBRC commands are issued.

**INST**

The INST keyword is optional on the GLBL command. This keyword is available in ICPSYSIN; it is also available in the global options module. Table 181 on page 477 describes the keyword.
The installation name value can be 1–39 alphanumeric characters. Surround the entire name with single quotation marks when spaces are included in the name. If you include spaces in the name, you must enclose the name in single quotation marks. If you enter the name in mixed upper/lower case (or for ISPF 2.3, in double-byte character sets such as Japanese-Kanji or Korean-Hanguel), your system printers must have the appropriate character sets in order to print the name correctly.

### Table 181 Image copy installation name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the name of your installation as you want it to appear in the heading portion of Image Copy function reports.</td>
<td>INST(instName)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> INST('BMC SOFTWARE, INC.')</td>
<td></td>
</tr>
</tbody>
</table>

The installation name value can be 1–39 alphanumeric characters. Surround the entire name with single quotation marks when spaces are included in the name. If you include spaces in the name, you must enclose the name in single quotation marks. If you enter the name in mixed upper/lower case (or for ISPF 2.3, in double-byte character sets such as Japanese-Kanji or Korean-Hanguel), your system printers must have the appropriate character sets in order to print the name correctly.

### IRLMGRP

The IRLMGRP keyword is valid with the REORG (Online Reorg function) and UNLOAD commands. This keyword is available in PLUSIN; it is not available as a global option. Table 182 describes the keyword.

### Table 182 IRLM group keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the IRLM group name.</td>
<td>IRLMGRP(value)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

Specifying a 1 to 8 alphanumeric character IRLM group name with the IRLMGRP keyword activates sysplex support. The name is passed to the Snapshot Copy function with the data set registration call. Snapshot technology uses this name to ensure that each active IRLM in a sysplex has an active subsystem. The registration call fails if the snapshot technology discovers an active IRLM without supporting the subsystem.

If IRLMGRP is specified, the databases must be defined in the RECONs with SHARELVL(3).
The KEYS keyword is valid with the OREORG command. You can specify this option in PLUSIN. Table 183 describes the keyword.

**Table 183  KEYS keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify root key values of records to be reorganized.</td>
<td>KEYS(value)</td>
</tr>
<tr>
<td>Internal default: none</td>
<td></td>
</tr>
</tbody>
</table>

The KEYS keyword is required if MODE(RECORD) is specified and the OLRKEYS DD statement is not specified. The KEYS keyword is optional when you specify an OLRKEYS DD statement. KEYS(value) is only applicable when MODE(RECORD) is specified.

The KEYS keyword specifies the root key values of records to be reorganized. You can specify up to 10 values. Each value is a character string with a maximum of 256 characters.

All characters are allowed, but any character string containing special characters must be enclosed in quotes (single or double). Single-quoted strings may be preceded by C (for character data) or X (for hexadecimal data). If X is used, the value inside the quotes must be an even number of characters and must consist only of characters 0 to 9 and A to F. Each value must be left justified and padded with X'FF's. An example follows:

```plaintext
KEYS(X'00000001',X'00000002')
```

If the OLRKEYS DD statement is specified, Online/Defrag processes the records in KEYS, then the records in OLRKEYS.

**LIM**

The LIM keyword is valid with the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or in PLUSIN. Table 184 on page 479 describes the keyword.
The Prefix Resolution/Update function notifies you when it finds orphan logical children or parents without children. This option determines how frequently to issue the notification. Specifying 500 will notify on the first, 501st, 1001st, and so on.

### Table 184  Limit between orphans keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notify user if orphan logical children or parents without children are found.</td>
<td>LIM(nnnn)</td>
</tr>
<tr>
<td>Do not notify user if orphan logical children or parents without children are found.</td>
<td>LIM () or do not specify keyword</td>
</tr>
<tr>
<td>Internal default: 500</td>
<td></td>
</tr>
</tbody>
</table>

The Prefix Resolution/Update function notifies you when it finds orphan logical children or parents without children. This option determines how frequently to issue the notification. Specifying 500 will notify on the first, 501st, 1001st, and so on.

### LIMITS

The LIMITS keyword is valid with the UNLOAD command. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 185 describes the keyword.

### Table 185  Parallel unload limits keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Divide the data into the specified ranges and unload.</td>
<td>LIMITS(startValue,endValue,ddname,...)</td>
</tr>
<tr>
<td>Allow the Unload function to divide the data as needed.</td>
<td>LIMITS(AUTO)</td>
</tr>
<tr>
<td>Use HALDB or the IMS/ESA Partition Support Product. Write one partition to each HD Unload file.</td>
<td>LIMITS(PART)</td>
</tr>
<tr>
<td>Internal Default: No division of data</td>
<td></td>
</tr>
</tbody>
</table>

The LIMITS keyword invokes the Parallel Unload facility, which allows you to split multivolume database data sets into subsets and unload the volumes concurrently. You can specify up to five subsets. If the data set extends across several volumes and much of the fragmentation of a database record is within the same volume (rather than fragmentation across several volumes), the Parallel Unload facility might run considerably faster.

**NOTE**

Do not use the Parallel Unload facility for databases that have secondary data set groups. It might degrade performance.
Table 186 shows keywords related to the LIMITS keyword.

Table 186 LIMITS keyword and other keywords

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Effect on Related Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARALLEL(YES)</td>
<td>Required with the LIMITS keyword.</td>
</tr>
<tr>
<td>PARALLEL(NO)</td>
<td>The LIMITS keyword is not valid with PARALLEL(NO). If you specify the LIMITS keyword and PARALLEL(NO), the Unload function issues an error message and terminates. For more information, see “PARALLEL” on page 504.</td>
</tr>
<tr>
<td>USEREXIT</td>
<td>not allowed in the same job step as LIMITS keyword</td>
</tr>
</tbody>
</table>

The LIMITS keyword works differently, depending on whether you are using the IMS/ESA Partition Support Product.

**HDAM and HIDAM databases**

This section explains how the Parallel Unload facility works for HDAM and HIDAM databases.

You can choose the ranges in the subsets by specifying starting and ending values, or you can allow the Unload function to choose the ranges by specifying LIMITS(AUTO).

You must include a DD statement for each subset. If you specify LIMITS(AUTO), the ddnames must be DFSURGU1, DFSURGU2, DFSURGU3, and so on. You can specify up to five data sets; the Unload function creates the corresponding number of subsets. For example, if you provide three DD statements, the Unload function splits the data into three subsets.

Each subset you define is unloaded by a subtask of the unload job. Each subtask allocates its own buffer pools. Each subtask initially allocates one cylinder of buffers for each data set group or partition in the database. The Unload function allocates more buffers as needed. When you use the Parallel Unload facility, increase the region size specified for the job to compensate for the additional buffers required.

An HD Unload data set is created for each subset. When you reload the database, concatenate these data sets together. The concatenated data sets appear to the Reload function or the IMS HD Reorganization Reload utility as a single data set.

HIDAM databases are split by root key ranges. HDAM databases are split by RAA block numbers. Use discretion in selecting these ranges to minimize DASD head contention.
If you do not specify LIMITS(AUTO), the LIMITS keyword must specify the starting and ending values for each range. You can specify the ddname for the unload data set; if you do not specify a ddname, the Unload function uses a default. An example of the LIMITS keyword follows:

\[
\text{LIMITS(} \text{startValue, } \text{endValue, } \text{ddname, } \ldots \text{)}
\]

The startValue, endValue, and ddname are required for each subset. You can specify up to five sets of values and/or subsets. Each startValue/endValue can be a maximum of 32 bytes. One DD statement is required for each subset defined; the DD statements specify the HD Unload data sets and must use the ddname specified in the LIMITS keyword. If these values are not specified, a ddname of DFSURGU1 is assumed for the first set of limits, a ddname of DFSURGU2 is assumed for the second set of limits, and so on.

**LIMITS keyword for HDAM**

In HDAM databases, the starting value must be a numeric value that specifies the RBN. The value must fall within the RAA. The first block, RBN 1, is the bitmap block. The unload begins at the first RAP in the specified RBN. If the starting value is not specified, a value of 1 is used.

The ending value must be a numeric value that specifies an RBN within the RAA. The unload terminates after the last RAP in the specified block. If the ending value is not specified, the last block in the RAA is used.

The examples shown in Figure 86 concurrently unload HDAM RBNs 1 through 500 to the DFSURGU1 data set and RBNs 501 through 1000 to the DFSURGU2 data set. If exactly 1,000 database records are in the database, each LIMITS keyword specifies the same subsets.

**Figure 86   Examples of JCL to unload HDAM RBNs**

\[
\begin{align*}
\text{LIMITS(1,500,DFSURGU1,501,1000,DFSURGU2)} \\
\text{LIMITS(1,500,,501,1000,)} \\
\text{LIMITS(,,500,501,)}
\end{align*}
\]

**LIMITS keyword for HIDAM**

In HIDAM databases, the starting and ending values must be specified in character or hexadecimal. If you specify hexadecimal, the hexadecimal string must contain an even number of hex characters. A key value length less than the root’s key length specifies a generic key. If you specify a generic key, the Unload function pads the starting key with X’00’s and the ending key with X’FF’s.
The unload begins at the first root with a key equal to or greater than the specified starting value. If you do not specify a starting value, the unload begins at the beginning of the database.

The unload terminates before processing the first root with a key greater than the ending value. If you do not specify an ending value, the unload continues to the end of the database.

The examples shown in Figure 87 assume the HIDAM root key is a 4-digit number. The database is being split into two subsets. All root keys of 4999 or less are written to the DFSURGU1 data set, and all root keys of 5000 or larger are written to the DFSURGU2 data set. Each example does the same thing.

**Figure 87  Examples of JCL to unload HIDAM**

<table>
<thead>
<tr>
<th>Limits</th>
<th>DFSURGU1</th>
<th>DFSURGU2</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIMITS('0000', '4999', DFSURGU1, '5000', '9999', DFSURGU2)</td>
<td>LIMITS('0', '4', DFSURGU1, '5', '9', DFSURGU2)</td>
<td>LIMITS(',') '4', ',', '5', ',', ) LIMITS('X'F4FF', DFSURGU1, 'X'F500', 'X'FFFF', DFSURGU2)</td>
</tr>
</tbody>
</table>

**HALDBs (PHDAM or PHIDAM) or databases using the IMS/ESA Partition Support Product**

This section explains how the Parallel Unload facility works for HALDBs or IMS/ESA Partition Support Product. You can unload selected partitions and take advantage of parallel processing by specifying the LIMITS keyword. If you are unloading a partitioned database, you cannot specify key ranges as limits; you must specify LIMITS(AUTO) or LIMITS(PART).

**LIMITS(AUTO)**

LIMITS(AUTO) works the same way for partitioned and non-partitioned databases. When you specify LIMITS(AUTO), the Unload function divides the database into subsets and the subsets are unloaded in parallel mode. You must specify the corresponding DD statements. For example, if you want to split the database into three subsets, specify three DFSURGUUn DD statements. You can specify up to five DFSURGUUn DD statements.

The Unload function splits the partitions into subsets by partition boundaries. For example, if your database has seven partitions and you specify four DFSURGUUn DD statements, the Unload function might split the database so that two partitions are in DFSURGU1, two partitions are in DFSURGU2, and so on.

The Unload function will never split a single partition into multiple subsets.
LIMITS(PART)

To unload selected partitions and use the Parallel Unload facility, specify LIMITS(PART) and PART(nn,nn,nn) or PART(partitionName, partitionName,...). For more information, see “PART (database partitioning)” on page 508.

If you are unloading selected partitions only (LIMITS(PART)), you cannot specify key ranges as limits. The Unload function uses the partitions themselves as the subsets.

Restrictions

The following restrictions apply when you use the LIMITS keyword:

- You cannot use the LIMITS keyword when you specify PARALLEL(NO).
- You cannot create dual HD Unload data sets with the DFSURGU2 DD statement.
- HISAM is not supported.
- Databases with compressed root keys are not supported.
- DATA PACKER/IMS version 2 and the DPKTLC exit available in DATA PACKER/IMS version 1 run successfully when you specify EXPAND(YES). If you use other compression routines, BMC recommends that you run the Parallel Unload facility with EXPAND(NO).

LOGDSN

The LOGDSN keyword is valid with the REORG (Online Reorg function) command. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 187 describes the keyword.

Table 187 Log data set name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify a data set that begins with the high-level qualifier bmc to record changes made to the database during the reorganization. Use a mask for the ddname. Use a single log.</td>
<td>LOGDSN(bmc.&amp;LOGDD, SINGLE)</td>
</tr>
</tbody>
</table>

Internal default: none

When you use the Online Reorg function, you can allow updates to the database while it is being reorganized. If any updates are made during the reorganization, the Online Reorg function records the updates and applies those updates to the reorganized database. As the Online Reorg function applies the updates, it creates an IEFRDER log.
The Online Reorg function modifies the IEFRDER log and creates a CRFRDER log that you can use if you need to recover the database after the reorganization. The IEFRDER and CRFRDER logs are required for each Online Reorg function job. You can specify these logs in the JCL, or you can have the Online Reorg function dynamically allocate them.

**NOTE**

To dynamically allocate dual logs, you must specify LOGDSN(&LOGDD).

Use the LOGDSN keyword to specify the name of the log that the Online Reorg function dynamically allocates. You must specify one IEFRDER-type log; you can specify two. For each IEFRDER log you specify, a corresponding CRFRDER log must be allocated. If you specify a generation data group (GDG), the data set name must be enclosed in single quotes; for example: LOGDSN('BMC.&LOGDD(+1')).

The LOGDSN keyword allows the following masks:

<table>
<thead>
<tr>
<th>Mask</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;DBD</td>
<td>DBD name of the database being reorganized</td>
</tr>
<tr>
<td>&amp;LOGDD</td>
<td>log ddname</td>
</tr>
<tr>
<td>&amp;DATE</td>
<td>log start date, in the format Dyyyyyddd</td>
</tr>
<tr>
<td>&amp;TIME</td>
<td>log start time, in the format Thhmmssst</td>
</tr>
</tbody>
</table>

You can specify any high-level qualifier.

The following example shows how you can use the masks. The high-level qualifier is BMC, and the DBD name is CUSTOMER.

```
LOGDSN(BMC.&DBD.&DATE.&TIME.&LOGDD)
```

The following data set name is built:

```
BMC.CUSTOMER.D1995209.T1232015.IEFRDER
```

When the Online Reorg function dynamically allocates the IEFRDER log, it deletes the IEFRDER log at the end of the job step. The CRFRDER log is kept for recovery purposes.
The LOGEXPDT and LOGRETPD keywords are valid with the REORG (Online Reorg function) command. You can specify the LOGRETPD option in the environment setup member (DLIGSET0) or in PLUSIN. The LOGEXPDT option is available in PLUSIN; it is not available as a global option. Table 189 describes the keywords.

Table 189  Log expiration date and retention period keywords

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keep the logs for nnnnn days.</td>
<td>LOGRETPD(nnnnn)</td>
</tr>
<tr>
<td>Keep the logs until the specified date.</td>
<td>LOGEXPDT(yyddd) or LOGEXPDT/yyyy/yyyy/ddd</td>
</tr>
</tbody>
</table>

Internal default: none

Use the LOGEXPDT and LOGRETPD keywords to specify how long to keep the logs. You can specify a number of days or an ending date.

If you specify an expiration date, you can use either of the following formats:

- **yyddd**
  
  Julian date format. For example, October 20, 1995 would appear as 95293.

- **yyyy/ddd**
  
  four-digit year and Julian-format days. For example, October 20, 1995 would appear as 1995/293.

You can specify LOGEXPDT or LOGRETPD. You cannot specify both keywords in the same job.

**LOGSMS**

The LOGSMS keyword is valid with the REORG (Online Reorg function) command. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 190 on page 486 describes the keyword.
If you are using SMS, use the LOGSMS keyword to specify the management class, storage class, and data class used to allocate the logs. The LOGSMS keyword value is determined by the value you specify in the environment setup member. If you do not specify a value for LOGSMS in the environment setup member, the value is determined by the value you specify for SMS in the environment setup member.

### LOGUNIT

The LOGUNIT keyword is valid with the REORG (Online Reorg function) command. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 191 describes the keywords.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the specified unit name to allocate the log.</td>
<td>LOGUNIT(unitName,unitName)</td>
</tr>
<tr>
<td>Internal default: Use the values specified in the installation parameters.</td>
<td></td>
</tr>
</tbody>
</table>

If you are not using SMS, use the LOGUNIT keyword to specify a unit name used to allocate the logs. The LOGUNIT keyword value is determined by the value you specify in the environment setup member. If you do not specify a value for LOGUNIT in the environment setup member, the value is determined by the value you specify for PERMUNIT in the environment setup member.

### LOGSPACE

The LOGSPACE keyword is valid with the REORG (Online Reorg function) command. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 192 on page 487 describes the keyword.
When you tell the Online Reorg function to dynamically allocate the logs, you can specify the number of cylinders to allocate for the primary and secondary extents. Use the LOGSPACE keyword to specify the amount of space you want.

**LPCK**

The LPCK keyword is valid with the DBSCAN, REORG (Reorg function), and UNLOAD commands. You can specify this option in an Unload function global options module or in PLUSIN. Table 193 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve logical parent concatenated keys (LPCKs).</td>
<td>LPCK(YES)</td>
</tr>
<tr>
<td>Bypass LPCK retrieval.</td>
<td>LPCK(NO)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> YES</td>
<td></td>
</tr>
</tbody>
</table>

Use the LPCK keyword to tell the Unload function not to extract the LPCK if the database contains logical child segments whose LPCKs are not physically stored in the segment (for example, they are virtual). Bypassing LPCK retrieval speeds up the unload or scan process. If you bypass LPCK retrieval with the unload function, you must specify DBR rather than DBIL for this DBD when running the IMS Prereorganization utility. DBR uses RBA pointers to match logical children and parents. DBIL uses symbolic keys to match logical children and parents.

Because a database containing a logical child segment with a direct logical parent (LP) pointer can be scanned only when the database containing the logical parent is reorganized with the DBR option, the DB Scan function always produces RBA pointers to match logical children with their logical parents.

When RBA pointers are used to match logical children and parents, LPCK retrieval detects bad logical parent (LP) pointers during the unload or scan process rather than later during prefix resolution. It will usually be faster to use POINTER CHECKER PLUS for the LP pointer checking before the reorganization rather than using LPCK retrieval. Because the POINTER CHECKER PLUS Hash Checking technique does not validate all occurrences of LP pointers in a twin chain, BMC recommends the Full Checking technique if LPCK retrieval is bypassed during the unload or scan.
If the LPCK is physically stored in the logical child segment, the Unload function ignores the LPCK keyword. Table 194 shows keywords related to the LPCK keyword.

### Table 194  LPCK keyword and other keywords

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Effect on related keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT(UNLDFILE)</td>
<td>The LPCK keyword is not valid and is ignored when you specify INPUT(UNLDFILE).</td>
</tr>
<tr>
<td>DBR in the Prereorganization utility for the Reorg function reorganization</td>
<td>The Reorg function uses LPCK(NO). You can override this on the REORG command.</td>
</tr>
</tbody>
</table>

You can set the default value for the LPCK option by using the Unload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

### MBI

The MBI keyword is valid with the OREORG and ANALYZE commands. You can specify this option in PLUSIN. Table 195 describes the keyword.

### Table 195  MBI keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the minimum number of expected blocks saved after reorganization to qualify the record for reorganization.</td>
<td>MBI(n)</td>
</tr>
<tr>
<td>Internal default: 5</td>
<td></td>
</tr>
</tbody>
</table>

The MBI (minimum block improvement) keyword specifies the minimum number of expected blocks to be saved per database record. For example, if MBI(5) is specified, the number of blocks that a record spans must be reduced by 5 after the reorganization. If the MBI criterion cannot be met, the record is not recommended for reorganization.

Valid values are 1 through 9999.

**NOTE**

BIPF() and MBI() criteria must be met to qualify a record for reorganization. For more information, see “BIPF” on page 413.
MINREC

The MINREC keyword is valid with the OREORG and ANALYZE commands. You can specify this option in PLUSIN. Table 196 describes the keyword.

Table 196 MINREC keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the minimum number of records to reorganize.</td>
<td>MINREC(n)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> variable (20% of the records that qualify for reorganization)</td>
<td></td>
</tr>
</tbody>
</table>

The MINREC (minimum records) specifies the minimum number of records to reorganize. The Online/Defrag Extractor and Recommendation Selector uses the MINREC value to calculate recommendation parameters. For example, if MINREC(50000) is specified, Online/Defrag calculates values for the BIPF and MBI keywords that will result in the greatest reduction of I/O activity.

Valid values are 1 through 9,999,999.

MODE

The MODE keyword is valid with the OREORG command. You can specify this option in PLUSIN. Table 197 describes the keyword.

Table 197 MODE keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reorganize selected database records.</td>
<td>MODE(RECORD)</td>
</tr>
<tr>
<td>Scan the entire database and reorganize records that meet specified criteria.</td>
<td>MODE(SCAN)</td>
</tr>
<tr>
<td>Reorganize all database records.</td>
<td>MODE(BLOCK)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

The MODE keyword is required.

The MODE keyword specifies the manner in which the database will be reorganized.

To reorganize specific database records identified by the OLRKEYS data set or the KEYS keyword, specify MODE(RECORD). A record will be reorganized if an analysis determines that it meets the criteria specified by the BIPF and MBI keywords.
To read the entire database and reorganize the database records that need to be reorganized, specify MODE(SCAN). A record will be reorganized if an analysis determines it meets the criteria specified by the BIPF and MBI keywords.

To read the entire database and reorganize all database records, specify MODE(BLOCK).

**MONITOR, MONUSERS, ROUTCODE, DESC CODE**

The MONITOR, MONUSERS, ROUTCODE, and DESC CODE keywords are valid with the BUILD, DBSCAN, RELOAD, REORG (Online Reorg function and Reorg function), UNLOAD, and OREORG commands. The DESC and ROUTCODE options are valid with the Prefix Resolution/Update function. You can specify the MONITOR option in the environment setup member (DLIGSET0) or in PLUSIN. The ROUTCODE and DESC CODE keywords are available in PLUSIN; they are not available as global options. Table 198 describes the keywords.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitor job progress at specified interval (issue message every nnnnnnnn seconds).</td>
<td>MONITOR(nnmmmm)</td>
</tr>
<tr>
<td>Send messages to TSO user ID.</td>
<td>MONUSERS(user01,user02,...)</td>
</tr>
<tr>
<td>Send messages to OS consoles.</td>
<td>ROUTCODE(n,n,n,...)</td>
</tr>
<tr>
<td></td>
<td>DESC(n,n,n,...)</td>
</tr>
<tr>
<td>Send messages to OS consoles. (Prefix Resolution/Update function)</td>
<td>ROUTCDE(n,n,n,...)</td>
</tr>
<tr>
<td></td>
<td>DESC(n,n,n,...)</td>
</tr>
<tr>
<td>Internal default: No automatic monitoring, send monitor messages to the console and BMCMSG, ROUTCODE(2,7), DESC CODE(7).</td>
<td></td>
</tr>
</tbody>
</table>

The MONITOR and MONUSERS keywords let you monitor the job. Monitoring is always available through the MVS MODIFY command; however, if you assign a value to the MONITOR keyword, MAXM Reorg solutions issue a message at a user-specified interval.
If you specify a TSO user ID with the MONUSERS keyword, MAXM Reorg/EP solutions send the messages to that TSO user’s terminal. You can specify as many as 10 user IDs. The user IDs are not checked for validity (for example, all messages sent to an invalid user ID are lost).

You can issue commands to request various monitoring tasks. To receive monitoring messages at specified intervals, you must specify a value for the MONITOR keyword.

The ROUTCODE and DESCCODE keywords allow you to specify routing and descriptor codes for MAXM Reorg solution messages. If you specify routing and descriptor codes, MAXM Reorg/EP solutions send the WTO to the MVS consoles indicated by the codes.

Routing codes send MAXM Reorg solution messages to the MVS system consoles where they can be displayed. More than one routing code can be assigned to a message to send it to more than one console. You can specify a maximum of 10 routing codes, separated by commas. The defaults are 2 (master console) and 7 (unit record pool). For the Prefix Resolution/Update function, the defaults are 1,7,11.

Descriptor codes describe the significance of messages. These codes determine how the system displays and deletes messages. You can specify a maximum of 10 descriptor codes, separated by commas. The default is 7.

For a description of message routing and descriptor codes, see the IBM MVS/ESA Routing and Descriptor Codes manual.

Use the MVS MODIFY command (F) to manipulate monitoring keywords. With the MODIFY command, you can show the monitor intervals and TSO user IDs in effect, display the progress of the job, and set or change values that were set with the MONITOR and MONUSERS keywords.

---

**WARNING**

If you are using an existing global options module that was created with the Database Utilities or MAXM Reorg classic, unpredictable results could occur. The Database Utilities and MAXM Reorg classic specified MONITOR(YES,nnnnn), where nnnnn represented a number of records. In MAXM Reorg/Online and the MAXM Reorg/EP solutions, the MONITOR keyword specifies an amount of time.

BMC recommends that if you use an existing global options module that specifies monitoring, you should override the monitoring option with a MONITOR(nnnnnnnn) keyword in the PLUSIN control statement data set. You can disable monitoring for the MAXM Reorg solutions by using the ISPF interfaces provided with the MAXM Reorg classic, but unpredictable results could occur.
The commands are described in the next section. Use the following format for all MODIFY commands, where jobName is the MVS job name and command is the specific utility command:

```
F jobName.command
```

### Show intervals and users in effect

To show the monitor intervals and TSO user IDs in effect, use the following command:

```
F jobName,STATUS
```

MAXM Reorg solutions issue messages similar to the following messages:

```
BMC90058I MONITOR INTERVAL IS 10 MIN 0 SEC
BMC90059I USERS TO RECEIVE MONITOR MESSAGES: (NONE)
```

### Display counters

To force a MAXM Reorg solution to display the current counters for the job being run, use the following command:

```
F jobName,DISPLAY
```

The function issues a message similar to the following message:

```
BMC90060 CUSTOMER DATABASE RECORDS UNLOADED 127
```

MAXM Reorg solutions send this message to the console, not to any TSO user IDs specified with the MONUSERS keyword. The DISPLAY command does not affect automatic monitoring that you specified with the MONITOR keyword.

### Set or change automatic monitoring values

To set or change automatic monitoring values that were set with the MONITOR and MONUSERS keywords, use the following command:

```
F jobName,I=aaaa,U=(bbb.ccc.ddd...)
```
NAME

In this command, I is the interval and can have the following values:

### Table 199 Interval values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aaa</td>
<td>automatic monitor interval in seconds</td>
</tr>
<tr>
<td>aaaS</td>
<td>automatic monitor interval in seconds</td>
</tr>
<tr>
<td>aaaM</td>
<td>automatic monitor interval in minutes</td>
</tr>
<tr>
<td>0</td>
<td>no automatic monitoring</td>
</tr>
</tbody>
</table>

U represents TSO user IDs and can have the following values:

### Table 200 User ID values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(bbb,ccc,ddd)</td>
<td>replace the current list (if any) with users bbb, ccc, and ddd</td>
</tr>
<tr>
<td>bbb</td>
<td>set a single user in the list</td>
</tr>
<tr>
<td>blank</td>
<td>empty the list</td>
</tr>
</tbody>
</table>

**NAME**

The NAME keyword is optional on the GROUP command. This keyword is available in ICPSYSIN; it is not available in global and database-specific options modules. Table 201 describes the keyword.

### Table 201 Image copy group name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify a unique identifier for a group that will be used by the Image Copy function.</td>
<td>NAME(grpName)</td>
</tr>
</tbody>
</table>

**Internal default:** NAME(CMDnnn)

If you use the NAME keyword, do not include the DBDSTRM keyword.

In the default value, NAME(CMDnnn), nnn is the command sequence number. The default applies only if you omit the DBDSTRM keywords.

For more information, see “GROUP control statement” on page 335.
NEWDBD

The NEWDBD keyword is valid with the UNLOAD commands. This keyword is available in PLUSIN; it is not available as a global option. Table 202 describes the keyword.

NOTE

The NEWDBD keyword is valid with the UNLOAD commands in the EP solutions and the utilities. It is not available with the classic solutions and utilities.

Table 202  Database restructure keyword (UNLOAD command)

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restructure the database.</td>
<td>NEWDBD(ddname)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: none</td>
<td></td>
</tr>
</tbody>
</table>

The NEWDBD keyword enables you to change a DBD during a reorganization. The NEWDBD keyword instructs the Unload function to restructure the database and specifies the ddname of the library that contains the new DBD definition.

If you change DBD values, you are responsible for ensuring that they are valid. For example, you can add or delete segments but you cannot change the hierarchical relationships of existing segments. Any changes to segment position within the database are subject to the guidelines in the IMS reorganization utilities. You should handle major changes to the structure outside of the reorganization, using a utility such as the POINTER CHECKER PLUS Data Restructure utility.

When you are unloading selected partitions of a partitioned database, the NEWDBD keyword is not valid. The NEWDBD keyword is valid only if you are unloading all partitions.

NOAUTH

The NOAUTH keyword is valid with the REORG (Reorg function) command. You can specify this option in a Reorg function global options module or in PLUSIN. Table 203 on page 495 describes the keyword.
The NOAUTH keyword specifies the action that the Reorg function takes if the database status in DBRC prohibits authorization. This situation occurs if you have issued a DBRC command like CHANGE.DB DBD(dbdName) NOAUTH or if you have issued a /DBR dbdName GLOBAL command.

If you intend to block DBRC authorization for everything except the reorganization by issuing the CHANGE.DB DBD(dbdName) NOAUTH command, use NOAUTH(RUN). The authorization process for the Reorg function ignores the DBRC NOAUTH flag you set.

If DBRC is not active, NOAUTH is ignored.

For more information, see “Database prohibits authorization” on page 105.

You can set the default value for the NOAUTH option by executing the batch job to create a Reorg function global options module. Sample JCL is in member #FRFGLBL of the sample library.

---

**Table 203  DBRC authorization keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continue Reorg function execution if database was set to NOAUTH.</td>
<td>NOAUTH(RUN)</td>
</tr>
<tr>
<td>Terminate Reorg function execution if database was set to NOAUTH.</td>
<td>NOAUTH(STOP)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> STOP</td>
<td></td>
</tr>
</tbody>
</table>

The NOAUTH keyword specifies the action that the Reorg function takes if the database status in DBRC prohibits authorization. This situation occurs if you have issued a DBRC command like CHANGE.DB DBD(dbdName) NOAUTH or if you have issued a /DBR dbdName GLOBAL command.

If you intend to block DBRC authorization for everything except the reorganization by issuing the CHANGE.DB DBD(dbdName) NOAUTH command, use NOAUTH(RUN). The authorization process for the Reorg function ignores the DBRC NOAUTH flag you set.

If DBRC is not active, NOAUTH is ignored.

For more information, see “Database prohibits authorization” on page 105.

You can set the default value for the NOAUTH option by executing the batch job to create a Reorg function global options module. Sample JCL is in member #FRFGLBL of the sample library.

---

**Table 204  Notification keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notify the specified user ID that a pertinent event has occurred.</td>
<td>NOTIFY(userId)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

Use this option to specify the TSO users to be notified when pertinent events occur within the Prefix Resolution/Update function.
The OLDDBD keyword is valid with the REORG (Online Reorg function and Reorg function) command. This keyword is available in PLUSIN; it is not available as a global option. Table 205 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change the DBD definition during the reorganization.</td>
<td>OLDDBD(ddname)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: none</td>
<td></td>
</tr>
</tbody>
</table>

The OLDDBD keyword enables you to change a DBD during a reorganization. The OLDDBD keyword specifies the ddname of the library containing the old DBD definition (the DBD that is used by the Unload function). The library defined by the IMS DD statement contains the new definition (the DBD that is used by the Reload function).

You can change certain DBD parameters when using the OLDDBD keyword:

- With the Online Reorg function, you can change only HDAM randomizer parameters. You might need to complete some tasks outside the reorganization. See “Online Reorg function” on page 498.

- With the Reorg function, you can change any DBD parameters except the following:
  - database organization
  - access method
  - number of data set groups
  - ddnames
  - hierarchy

- SWAP(DELETE) is forced to SWAP(YES) if you specify OLDDBD and shorten the segment length.

If you need to change DBD parameters not supported by the OLDDBD keyword, reorganize the database using stand-alone utilities.
When you change DBD values, you are responsible for ensuring that the new values are valid. You should handle major changes to database structure outside of the reorganization, using a utility such as the POINTER CHECKER PLUS Data Restructure utility.

The Reorg function DBD Change Summary report is generated when the OLDDBD keyword is used. This report shows the new DBD definition with any fields from the old definition that have changed.

If you specify OLDDBD(IMS), the OLDDBD keyword is ignored.

**HDAM randomizer changes**

When you specify the OLDDBD keyword, the utility checks whether any of the following HDAM randomizer parameters were changed:

- number of RAPs
- number of RAA blocks
- randomizing module name

Because the HD Sort utility must be able to be executed when you change any randomizer parameters except maximum insert bytes, you must specify HDSORT(YES) or HDSORT(AUTO) with the OLDDBD keyword. (The utility determines whether it needs to invoke the HD Sort utility or the BMC RAP Sequencer.) If you specify HDSORT(NO) with the OLDDBD keyword, the reorganization utility will not be able to invoke the HD Sort utility. An error message will be issued later if the HD Sort utility was needed and you specified HDSORT(NO). For more information, see “HDSORT” on page 443.

**Reorg function**

When you change DBD values, you are responsible for ensuring that the new values are valid. For example, you can add or delete segments but you cannot change the hierarchical relationships of existing segments. Any changes to segment position within the database are subject to the guidelines in the IMS reorganization utilities.

If DBRC is active and you specify SWAP(YES) or SWAP(D), the Reorg function turns on the PROHIBIT AUTHORIZATION flag for the database and writes message BMC90086 to the MVS console and BMCMSG. This allows you to perform other tasks (such as running the ACBGEN or updating DBD libraries) before anyone can access the database. You must run the DBRC batch utility (DSPURX00) to turn this flag off. (Issue the CHANGE.DB DBD(dbdName) AUTH command.)

If the database has compressed segments and you need those segments expanded, you must specify EXPAND(YES). The Reorg function default is EXPAND(NO). For details, see “COMPRESS, EXPAND” on page 415.
The OLDDBD keyword is not valid when you specify INPUT(UNLDFILE). If you want to change the randomizer parameters for an HDAM database, specify HDSORT(YES) or HDSORT(AUTO) and use the IMS DD statement to specify the DBD with the new parameters.

The OLDDBD keyword is not valid in a one step reorg for HALDB.

Online Reorg function

You can change some DBD parameters during an Online Reorg function reorganization by specifying the OLDDBD keyword in the reorganization job. When you change DBD values, you are responsible for ensuring that the new values are valid.

The following parameters can be changed:

- randomizer name
- number of RAPs
- RAA size
- maximum insert bytes
- free space (FRSPC)
- physical block size or CISIZE (using snapshot feature only)

If you need to change other DBD parameters, use the Reorg function or use individual functions to reorganize the database.

Be careful when using the OLDDBD and SWAP(YES) keywords during an Online Reorg function reorganization; changing the DBD will have an effect on your IMS online systems. The effect depends on whether you have DELTA IMS DB/DC installed. You might need to complete tasks after the reorganization. For more information, see “DBD changes during reorganization” on page 129.

Secondary index changes

You can implement DBD changes to secondary indexes during a Reorg function reorganization. This section describes tasks you must complete outside of the reorganization job.
Deleting secondary indexes

To delete a secondary index, you do not need to take any special action before the Reorg function reorganization job, other than changing the DBD. Because the deleted secondary index will not be created during the reorganization job, its data set name will not be swapped. Delete the secondary index from your system after the reorganization by removing it from the following:

- DBRC
- DBDLib
- ACBLIB
- database data sets

Adding secondary indexes

You can add secondary indexes during a Reorg function reorganization job. Because complete DBRC control is required for data set name swapping, you must complete the following tasks before you submit the reorganization job:

1. Define the new secondary index to DBRC by using the DSPURX00 utility. Use the INIT.DB and INIT.DBDS functions.

2. Allocate secondary index data sets as follows:

   - If you specified SHARE(NO), allocate the new secondary index data set by using IDCAMS.

   - If you specified SHARE(YES), allocate a shadow secondary index data set for the new secondary index to be written to. Allocate another empty data set to be used for the data set name swapping process, as registered to DBRC.

OSAMINIT

The OSAMINIT keyword is valid with the RELOAD, REORG, GROUP, and USERLOAD commands to control the action the utility takes if the database has multi-volume OSAM data sets that are not managed by IBM’s DFSMS product. You can specify this option in PLUSIN. Table 206 on page 500 describes the keyword.
When OSAMINIT(NO) is specified, DLILFREN bypasses the call to DLIGVTOC and multi-volume OSAM data sets do not have the high-used RBA reset in the VTOC.

### Table 206 OSAMINIT database keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the database has multi-volume OSAM data sets that are not managed by IBM’s DFSMS product, reset the high-used RBA to eliminate the possibility of having a false EOF left in the DSCB of the last volume of a previously existing data set.</td>
<td>OSAMINIT (YES)</td>
</tr>
<tr>
<td>Do not set the high-used RBA.</td>
<td>OSAMINIT (NO)</td>
</tr>
<tr>
<td>Internal default: YES</td>
<td></td>
</tr>
</tbody>
</table>

The OSAMMAX keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. For the Reload function and the Reorg function, this keyword is available in PLUSIN; it is not available as a global option. For the Online Reorg function, you can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 207 describes the keyword.

### Table 207 8-GB OSAM database keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reload an OSAM database that is larger than 4 GB.</td>
<td>OSAMMAX(8)</td>
</tr>
<tr>
<td>Reload an OSAM database that is smaller than 4 GB.</td>
<td>OSAMMAX(4)</td>
</tr>
<tr>
<td>Internal default: 4</td>
<td></td>
</tr>
</tbody>
</table>

The OSAMMAX keyword allows you to reload an OSAM database that is larger than 4 GB. The OSAMMAX keyword is used only for IMS Version 5.1. If you are running IMS 4.1 or earlier, databases can be no larger than 4 GB. If you are running IMS 6.1, 8 GB databases are allowed.

**NOTE**

This option is used only if you are using IMS 5.1 and have applied IBM APAR PN82671, which enables you to use OSAM databases larger than 4 GB. If you have not applied PTF PN82671 to your IMS 5.1 and you specify OSAMMAX(8), you might attempt to load a database that cannot be processed by IMS.

To use an 8-GB OSAM database, the block size must be even.
You can change the default to be OSAMMAX(8) by executing the SETUP macro. You can execute the macro when you install the product or at any time you want to change system-level defaults. Member #DLIGLBL of the sample library contains the SETUP macro.

**OUTDD**

The OUTDD keyword is valid with the COPY command. This keyword is available in PLUSIN; it is not available as a global option. Table 208 describes the keyword.

**Table 208  OUTDD keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use this keyword to identify the DD statement for the copied database.</td>
<td>OUTDD(value)</td>
</tr>
<tr>
<td>Internal default: none</td>
<td></td>
</tr>
</tbody>
</table>

The OUTDD keyword specifies the DD statement for the copied database. For HALDBs, use the keyword to specify where to write the copied partition.

Valid values are from one through eight alphanumeric characters.

OUTDD cannot be used to copy an ILDS of a HALDB; the ILDS is automatically copied.

**NOTE**

If you use INDD/OUTDD, you cannot use dynamic allocation. BMC recommends dynamic allocation.

**OUTDDX**

The OUTDDX keyword is valid with the RELOAD and COPY commands. This keyword is available in PLUSIN; it is not available as a global option. Table 209 on page 502 describes the keyword.
**OUTDDX**

The OUTDDX keyword enables you to place secondary index records in a data set other than DFSURWF1. When the OUTDDX keyword is specified, secondary index records are not written to the DFSURWF1 data set; the records for each secondary index are written to separate data sets.

If you do not include an OUTDDX keyword, all secondary index records are written to the DFSURWF1 data set or HLDURIDX.

The OUTDDX keyword does not support shared secondary indexes.

The Reload function OUTDDX keyword is normally used with the Index Build function INDDX keyword.

### OUTDDX(ddname1(…))

Specify the output ddnames to use and the secondary index WF1 records to be written to them in the following format (maximum of 256 characters):

\[
\text{OUTDDX(} \text{ddname1(indexdb1), ddname2(indexdb2)} \ldots \text{)}
\]

You must specify each secondary index, and each secondary index must have a different ddname. The records for any secondary index not specifically selected are written to DFSURWF1.

### OUTDDX(AUTO)

When you specify OUTDDX(AUTO), control records are written to the DFSURWF1 file along with DFSURWF1 records for logical relationships. All secondary index records are written to separate DFSURWF1-format data sets.

If you specify INDDX(AUTO) in a subsequent Index Build function job step, the file written to DFSURWF1 in the Reload function step is used to dynamically allocate all the separate OUTDDX DFSURWF1 files.

---

**Table 209  Write secondary index records to non-DFSURWF1 file keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Place secondary index work file records in a data set other than DFSURWF1.</td>
<td>OUTDDX(ddname1(indexdb1), ddname2(indexdb2)...)</td>
</tr>
<tr>
<td>Place secondary index work file records in dynamically allocated data sets other than DFSURWF1.</td>
<td>OUTDDX(AUTO)</td>
</tr>
</tbody>
</table>

Internal default: none
To use OUTDDX(AUTO), you must include the following DD statements to the JCL:

- one DD statement for each secondary index. The ddname must match the secondary index DBD name.
- one DD statement with a ddname of DFSURWF1

### OVFLONLY

The OVFLONLY keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. This keyword is available in PLUSIN; it is not available as a global option. Table 210 describes the keyword.

#### Table 210  Overflow only keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Place specified segments in the HDAM or PHDAM overflow area.</td>
<td>OVFLONLY(segName,segName,...)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

The OVFLONLY keyword allows you place specified segments into the HDAM or PHDAM overflow area. You can place up to 25 segments in the overflow area.

If you specify the OVFLONLY keyword and the SEGPOS keyword in the same job step, the SEGPOS keyword is ignored.

### PADCHAR

The PADCHAR keyword is valid with the RELOAD and REORG (Online Reorg function and Reorg function) commands. You can specify this option in PLUSIN. Table 211 describes the keyword.

#### Table 211  PADCHAR keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the character to use to pad segments that are extended between the unload and reload.</td>
<td>PADCHAR(x) PADCHAR('x') PADCHAR(&quot;x&quot;) PADCHAR(C'x') PADCHAR(X'xx') PADCHAR(X'00')</td>
</tr>
<tr>
<td><strong>Internal default:</strong> PADCHAR(X'00')</td>
<td></td>
</tr>
</tbody>
</table>
Specify PADCHAR values according to the following rules:

- The PADCHAR value may be any character, but special characters must be enclosed in quotation marks (single or double).
- Single quotation marks may be preceded by a C or X to denote character or hexadecimal data, respectively.
- If X is used, the value inside the quotation marks must be an even number and must consist of the alphanumeric characters 0-9 and A-F.

**PARALLEL**

The PARALLEL keyword is valid with the UNLOAD, BUILD, RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 212 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process the files in parallel mode.</td>
<td>PARALLEL(YES)</td>
</tr>
<tr>
<td>Do not process the files in parallel mode.</td>
<td>PARALLEL(NO)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> YES</td>
<td></td>
</tr>
</tbody>
</table>

For best performance, BMC recommends that you specify PARALLEL(YES), or accept it as the default, whenever possible. The PARALLEL(YES) specification enables parallel processing but does not explicitly force parallel processing. When a utility finds PARALLEL(YES) in a job step (whether it is accepted as a default or specified in the PLUSIN control statement data set), the utility will process the files in parallel mode when that is the most efficient processing method.

**Limitations**

The PARALLEL keyword has the following limitations:

- In the Unload function and the Reload function, HISAM and SHISAM databases are forced to PARALLEL(NO).
- In the Unload function and the Reload function, if you specify a USEREXIT keyword databases are unloaded or reloaded in non-parallel mode.
In the Online Reorg function and the Reorg function, HISAM and SHISAM databases are unloaded and reloaded in non-parallel mode. Secondary indexes are built in parallel.

In the Online Reorg function and the Reorg function when you specify ICP(YES), all database types are unloaded and loaded in non-parallel mode. Secondary indexes are built in parallel.

If you specify PARALLEL(NO) in the Index Build function, the Online Reorg function, or the Reorg function, secondary indexes are built in non-parallel mode (one at a time).

For the Reorg function, you can specify PARALLEL(YES) with the USEREXIT keyword; however, parallel processing will be disabled for the primary database. The secondary indexes will be built in parallel mode.

### Unload function

The PARALLEL keyword enables you to split a database and unload the subsets in parallel (at the same time). You can specify as many as five subsets. Because you are unloading multiple subsets at the same time, you save elapsed time.

If you want to unload the database in subsets, you must specify PARALLEL(YES) (or accept it as the default) and include the LIMITS keyword to specify the number and ranges of the subsets. You can specify up to five subsets, and you must supply ddnames for each output data set. Include the appropriate DD statements (DFSURGU1, DFSURGU2, and so on) in the JCL. For example, if you specify three subsets with the LIMITS keyword, you must supply ddnames for three output data sets. For more information, see “LIMITS” on page 479.

If you want two copies of the unload data set, you must specify PARALLEL(NO). DFSURGU2 contains the copy data set for the unload. If you specify PARALLEL(NO), do not specify the LIMITS keyword; the system will return an error.

If a database is unloaded in parallel mode, it can be loaded in parallel mode or non-parallel mode. For a database to be loaded in parallel mode, it must be unloaded in parallel mode.

### Reload function

The PARALLEL keyword enables you to split a database and load the subsets in parallel (at the same time). You can specify as many as five subsets. Because you are loading multiple subsets at the same time, you save elapsed time.
To reload a database in parallel mode, complete the following steps:

1. Unload the database with the Unload function. Use PARALLEL(YES) and the LIMITS keyword. Specify one DFSURGUn DD statement for each subset in the JCL.

2. In the Reload function, specify PARALLEL(YES) or accept it as the default.

3. Specify input DFSUINnn DD statements in the JCL. The Reload function checks the JCL to see how many input files exist. The DD statements must be in the correct order.

   For example, the unload file DFSURGU4 must be the input file DFSUIN04:

   - If you specify multiple DFSUINnn DD statements (DFSUIN01, DFSUIN02, and so on) and PARALLEL(YES), the database will be reloaded in parallel mode.

   - If you specify DFSUINnn DD statements in the JCL but you specify PARALLEL(NO), the Reload function reloads the DFSUIN01 data set, then the DFSUIN02 data set, and so on.

### Index Build function

The PARALLEL keyword enables you to split WF1 records and to sort and load them in parallel mode (at the same time). Because you are sorting and loading multiple index data sets at the same time, you save elapsed time.

When you specify PARALLEL(YES) with the BUILD command, the Index Build function splits the work file data set (or equivalent) into the optimum number of data sets (as many as three). The Index Build function sorts the index records and builds the indexes simultaneously.

When you use a hierarchical scan and PARALLEL(YES), the Index Build function scans the database in parallel mode. The Index Build function chooses the optimum number of scan tasks (up to five).
Reorg function

When you specify PARALLEL(YES) and do not specify ICP(YES) with the REORG command, the database can be unloaded and reloaded and secondary indexes can be built in parallel mode. The Reorg function decides whether to use parallel processing, based on the size of the primary data set or HDAM or PHDAM RAA. The internal default divides the database into 2000-cylinder increments.

When you specify PARALLEL(YES) and ICP(YES) with the REORG command, secondary indexes can be built in parallel mode. The database will be unloaded and reloaded in non-parallel mode.

When you specify PARALLEL(NO) with the REORG command, parallel processing is disabled for all utilities.

Online Reorg function

When you specify PARALLEL(YES) with the REORG command, only secondary indexes can be built in parallel mode. The database will be unloaded and reloaded in non-parallel mode.

When you specify PARALLEL(NO) with the REORG command, parallel processing is disabled for all utilities.

PARMBLK

The PARMBLK keyword is optional on the GLBL and REORG commands. This keyword is available in ICPSYSIN; it is also available in the global options module. Table 213 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify the name assigned to the global options module to use for the Image Copy function.</td>
<td>PARMBLK(modName)</td>
</tr>
<tr>
<td>Internal default: PARMBLK(ICPPARMS)</td>
<td></td>
</tr>
</tbody>
</table>

Use this keyword to specify a different global options module to use. The module name supplied with this keyword must have been created using the customization process for the Image Copy function. For more information, see “Global options modules” on page 280.

The module must be named ICPPARMS or ICP@xxx, where xxx can be up to four alphanumeric or national characters (A–Z, 0–9, @#$).
PART (database partitioning)

The PART keyword is valid with the UNLOAD, BUILD, RELOAD, REORG, OREORG, and GROUP commands. This keyword is available in PLUSIN; it is not available as a global option. Table 214 describes the keyword.

Table 214  Partitioning keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process selected partitions. (IMS/ESA Partition Support Product and HALDB)</td>
<td>PART(partitionName, ...)</td>
</tr>
<tr>
<td>Process selected partitions. (IMS/ESA Partition Support Product only)</td>
<td>PART(nn,nn,...)</td>
</tr>
<tr>
<td>Process all partitions. (IMS/ESA Partition Support Product and HALDB)</td>
<td>PART(ALL)</td>
</tr>
<tr>
<td>Process a selected partition. (Online/Defrag)</td>
<td>PART(nn)</td>
</tr>
</tbody>
</table>

**Internal default: ALL**

If a HALDB uses a Partition Selection Exit routine that returns a subset of the partitions, the internal default of ALL processes the partitions returned by the Partition Selection Exit routine. To process all the partitions when a Partition Selection Exit routine is used, explicitly specify PART(ALL) in PLUSIN.

The PART keyword is valid for HALDBs. Currently, you can specify PART for the Unload function, Reload function, Index Build function, and Reorg function, including DEFRAG. Using the PART keyword, you can build primary indexes and Indirect List Data Sets (ILDS) for selected HALDB partitions.

**NOTE**

You can also build secondary indexes for selected HALDB partitions by using the SECINDEX keyword. For more information, see “HALDBs” on page 259.

MAXM Reorg solutions support the IMS/ESA Partition Support Product, which allows you to split a database into as many as 32 partitions. With the MAXM Reorg/EP or MAXM Reorg/Online for IMS solutions, you can reorganize selected partitions. The PART keyword enables partitioning support. Because the default is PART(ALL), you need to specify the PART keyword in the PLUSIN control statement data set only if you want to process selected partitions.
If you are performing reorganization tasks for selected partitions, the following restrictions apply:

- The USERHDR keyword is not supported.
- You cannot make DBD changes during the reorganization.
- If the partitioned database has logical relationships, you must specify DBR (rather than DBIL) when creating the control data set (CDS) with the IMS Prereorganization utility.

If you perform reorganization tasks for selected partitions, rather than the entire database, you must make JCL changes as described in the following sections.

**Unload function and Reload function**

When you specify the PART keyword, you can unload all partitions or selected partitions. If you accept the default of PART(ALL), all partitions are unloaded.

**NOTE**

If a HALDB uses a Partition Selection Exit routine that returns a subset of the partitions using “Select Next Partition” processing, the internal default of ALL restricts the partitions processed to the subset returned by the Partition Selection Exit routine. Explicitly specifying PART(ALL) in PLUSIN will process all the partitions.

When you specify the PART keyword, you can reload all partitions or selected partitions. If you accept the default of PART(ALL), all partitions are reloaded. The PART keyword is available with the RELOAD command when using an IBM unload file as input or a partial reload of an IBM partitioned database. If there is any input to a partition that is not specified in the PART option, the reload is aborted. When processing a BMC unload file, this keyword may not be used.

If you specify PART(nn,nn,nn) and the database does not use the IMS/ESA Partition Support Product, an error message is issued.

The only way to create a second copy of the HD Unload file for a database that uses the IMS/ESA Partition Support Product is to specify PARALLEL(NO) and a DFSURGU2 DD statement with the DFSURGU1 DD statement. Do not specify the LIMITS keyword when you specify PARALLEL(NO); the system will return an error.

To unload the database in parallel mode, you must specify PARALLEL(YES) and the LIMITS keyword with the PART keyword. If you are unloading a HALDB or a database that uses the IMS/ESA Partition Support Product, you cannot specify key ranges as limits; you must specify LIMITS(AUTO) or LIMITS(PART). For more information, see “LIMITS” on page 479.
To process subsets of the database that are not individual partitions, use the Database Selection facility (DBSF). You must know the boundaries of the subsets. For example, if you want to unload the last part of partition 1 and the first part of partition 2, you can use the DBSF to extract that subset with the Unload function. For more information, see “Database Selection facility” on page 192.

The `partitionName` value must be a valid HALDB partition name for the database being unloaded. The unload file ddnames for HALDB partitions are constructed by appending a 1 to the partition name. For duplicate copies, ddnames are constructed appending a 2 to the partition name.

For the Unload function, the values of the LIMITS and PART keywords determine the contents of the HD Unload files and the required ddnames. The DD statements that you specify determine how many subtasks will be used.

For the Reload function, the value of the PARALLEL keyword and the input DD statements determine how many subtasks will be used. If you specify a DFSUINPT DD statement, you must concatenate all HD Unload data sets in the correct order. If you specify DFSUINnn DD statements, the Reload function DD statements must match the Unload function DD statements. You must reload all files that were unloaded; you cannot reload a subset of what you unloaded.

BMC recommends that, for best performance, you reload the data sets in parallel mode (specify PARALLEL(YES) or accept it as the default). If you are reloading multiple subsets of a HIDAM or PHIDAM partition and you try to execute multiple, separate reload jobs at the same time, the reload jobs will abend. The Reload function requires exclusive control of the primary index.

If you unload selected partitions and place more than one partition in an HD Unload file, you must specify PARALLEL(YES) when you reload the partitions in the Reload function.

**NOTE**
Currently, you cannot build secondary indexes for selected partitions of a PHDAM or PHIDAM database.

**Reloading a subset of the partitions**

Normally, the Reload function uses information in the unload file header to chose which partitions are reloaded. In normal cases, the PART keyword does not need to be coded in the Reload jobstep. The following exceptions apply:

- If the unload was performed by a non-BMC utility, then the header records in the unload file do not contain the information needed to chose the partitions to be reloaded. In this case, the user must code the PART keyword in the Reload step.
- If an existing HALDB is having its partition boundaries changed, the information in the unload file header is ignored. In this case, it is possible to unload and reload the entire HALDB. However, unloading and reloading the entire HALDB is not always practical. When changing boundaries during a partial partition Unload/Reload, ensure that all database records in the new partitions ranges are unloaded and fed into the Reload job step. If a partition that was not unloaded contains database records within the new keyrange limit of a partition built during the reload, those records are not available for random retrieval. Furthermore, if an unload file is omitted from the reload, then its database records are dropped. Neither of these errors are detected by the Reload job step.

- A user might want to manually code the PART keyword in a Reload job step during some Full Function to HALDB conversions. Most Full Function to HALDB conversions should be run without specifying the PART keyword. An exception to this is where a existing application level partitioning structure is being converted into a HALDB. For example, separate Full Function databases may be used to hold records based on a key range and this entire set of Full Function databases might be converted to partitions of a HALDB on a one for one basis. In this case, a separate unload file might be fed into a separate Reload job step with the PART keyword used to identify the partition intended to receive those database records. The unload files header records cannot be used to ensure that all records are directed to the correct partition.

In all these cases the Reload function ensure that each database record read from the unload file is loaded into a partition that matches its key.

Table 215 shows examples of unloading and reloading all or selected partitions.

**Table 215  LIMITS and PART keyword combinations (part 1 of 2)**

<table>
<thead>
<tr>
<th>Unload function keywords</th>
<th>Unload function DD statements</th>
<th>Reload function DD statements</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART(ALL) LIMITS(AUTO)</td>
<td>DFSURGUn (up to 5)</td>
<td>DFSUINPT (all files concatenated) or DFSUINnn (up to 5)</td>
<td>All partitions are unloaded. A maximum of five unload files are allowed with LIMITS(AUTO). If you are unloading more partitions than you have DD statements for, some data sets will contain multiple partitions.</td>
</tr>
<tr>
<td>PARALLEL(YES) is required</td>
<td>One unload subtask per unload file</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PART(n,n,n) LIMITS(AUTO)</td>
<td>DFSURGUn (up to 5)</td>
<td>DFSUINPT (all files concatenated) or DFSUINnn (up to 5)</td>
<td>The partitions you specified are unloaded. A maximum of five unload files are allowed with LIMITS(AUTO). If you are unloading more partitions than you have DD statements for, some data sets will contain multiple partitions.</td>
</tr>
<tr>
<td>PARALLEL(YES) is required</td>
<td>One unload subtask per unload file</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 215  LIMITS and PART keyword combinations (part 2 of 2)

<table>
<thead>
<tr>
<th>Unload function keywords</th>
<th>Unload function DD statements</th>
<th>Reload function DD statements</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART(ALL) LIMITS(PART) PARALLEL(YES) is required</td>
<td>DFSURGnn (up to 32)</td>
<td>DFSUINPT (all files concatenated) or DFSUINnn (up to 32)</td>
<td>All partitions are unloaded. All data sets unloaded must be input into the reload. Each file has a BMC header record; only the last file has a trailer record. (This syntax is only for non-HALDB databases.)</td>
</tr>
<tr>
<td>PART(n, n) LIMITS(PART) PARALLEL(YES) is required</td>
<td>DFSURGnn (one DD statement for each partition to be unloaded)</td>
<td>DFSUINPT (all files concatenated) or DFSUINnn (one for each partition)</td>
<td>All data sets used in the unload must be input into the reload; you cannot reload a subset of what was unloaded. The suffix of each file name must correspond to the partition number.</td>
</tr>
<tr>
<td>PART(ALL) no LIMITS keyword</td>
<td>DFSURGU1 DFSURGU2 can only be a second copy of HD Unload file.</td>
<td>DFSUINPT</td>
<td>All partitions are unloaded and reloaded using one file.</td>
</tr>
<tr>
<td>PART(n, n) no LIMITS keyword</td>
<td>DFSURGU1 DFSURGU2 can only be a second copy of HD Unload file.</td>
<td>DFSUINPT</td>
<td>Selected partitions are unloaded and reloaded.</td>
</tr>
<tr>
<td>PART(ALL) LIMITS(PART) PARALLEL(YES) is required</td>
<td>part1name1 part2name1 part3name1 part4name1</td>
<td>part1name part2name part3name part4name</td>
<td>All partitions are unloaded. All data sets unloaded must be input into the reload. Each file has a BMC header record; only the last file has a trailer record. Valid for HALDBs.</td>
</tr>
<tr>
<td>PART(ALL) LIMITS(PART) PARALLEL(YES) is required</td>
<td>part1name1 part2name1 part3name1 part4name1</td>
<td>DFSUINPT (all files concatenated)</td>
<td>All partitions are unloaded. All data sets unloaded must be input into the reload. Each file has a BMC header record; only the last file has a trailer record. Valid for HALDBs.</td>
</tr>
<tr>
<td>PART(PART1NAME, PART2NAME) LIMITS(PART) PARALLEL(YES) is required for Unload function</td>
<td>part1name1 part2name1</td>
<td>part1name part2name</td>
<td>The partitions you specified are unloaded. All data sets used in the unload must be input into the reload; you cannot reload a subset of what was unloaded.</td>
</tr>
<tr>
<td>PART(PART1NAME, PART2NAME) LIMITS(PART) PARALLEL(YES) is required for Unload function</td>
<td>part1name1 part2name1</td>
<td>DFSUINPT (all files concatenated)</td>
<td>The partitions you specified are unloaded. All data sets used in the unload must be input into the reload; you cannot reload a subset of what was unloaded.</td>
</tr>
</tbody>
</table>
Table 216 shows examples of a database that uses HALDB partitioning.

<table>
<thead>
<tr>
<th>Unload function keywords</th>
<th>Unload function DD statements</th>
<th>Reload function DD statements</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART(ALL) LIMITS(AUTO) PARALLEL(YES) is required</td>
<td>DFSURGU1 DFSURGU2 DFSURGU3 Three unload subtasks</td>
<td>DFSUIN01 DFSUIN02 DFSUIN03 Three reload subtasks</td>
<td>The unload DD statements must match the reload DD statements. The LIMITS(AUTO) value determines which partitions are written to which HD Unload data sets.</td>
</tr>
<tr>
<td></td>
<td>DFSURGU1 DFSURGU2 DFSURGU3 Three unload subtasks</td>
<td>DFSUINPT file1 file2 file3 One reload subtask</td>
<td>DFSUINPT requires that all HD Unload files be concatenated in the correct order.</td>
</tr>
<tr>
<td>PART(ALL) LIMITS(PART) PARALLEL(YES) is required</td>
<td>part1name1 part2name1 part3name1 part4name1 part5name1 Five or less unload subtasks</td>
<td>part1name part2name part3name part4name part5name Five or less reload subtasks</td>
<td>All partitions are unloaded. All data sets unloaded must be input into the reload. Each file has a BMC header records; only the last file has a trailer record. If partXname2 ddnames are specified for the Unload, two copies of the unload file are created.</td>
</tr>
<tr>
<td></td>
<td>part1name1 part2name1 part3name1 part4name1 part5name1 Five or less unload subtasks</td>
<td>DFSUINPT file1 file2 file3 One reload subtask</td>
<td>DFSUINPT requires that all HD Unload files be concatenated in the correct order.</td>
</tr>
<tr>
<td>PART(PART1NAME, PART3NAME) LIMITS(PART) PARALLEL(YES) is required</td>
<td>part1name1 part3name1</td>
<td>part1name part3name Parallel load will not be supported</td>
<td>Partitions 1 and 3 are unloaded. All data sets unloaded must be input into the reload.</td>
</tr>
<tr>
<td>PART(PART1NAME, PART3NAME) LIMITS(PART) PARALLEL(YES) is required</td>
<td>part1name1 part3name1</td>
<td>DFSUINPT file1 file3 One reload subtask</td>
<td>DFSUINPT requires that all HD Unload files be concatenated in the correct order.</td>
</tr>
</tbody>
</table>
Table 217 shows examples of a database that uses the IMS/ESA Partition Support Product.

### Table 217 IMS/ESA partition support product examples (part 1 of 2)

<table>
<thead>
<tr>
<th>Unload function keywords</th>
<th>Unload function DD statements</th>
<th>Reload function DD statements</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART(ALL) or PART(4,6,7) and LIMITS(AUTO) PARALLEL(YES) is required</td>
<td>DFSURGU1 DFSURGU2 DFSURGU3</td>
<td>DFSUIN01 DFSUIN02 DFSUIN03</td>
<td>The unload DD statements must match the reload DD statements. The LIMITS(AUTO) value determines which partitions are written to which HD Unload data sets.</td>
</tr>
<tr>
<td></td>
<td>Three unload subtasks</td>
<td>Three reload subtasks</td>
<td>PARALLEL(YES) is required</td>
</tr>
<tr>
<td>PART(ALL) or PART(4,6,7) and LIMITS(AUTO) PARALLEL(YES) is required</td>
<td>DFSURGU1 DFSURGU2 DFSURGU3</td>
<td>DFSUINPT file1 file2 file3</td>
<td>DFSUINPT requires that all HD Unload files be concatenated in the correct order.</td>
</tr>
<tr>
<td></td>
<td>Three unload subtasks</td>
<td>One reload subtask</td>
<td></td>
</tr>
<tr>
<td>PART(ALL) LIMITS(PART) PARALLEL(YES) is required</td>
<td>DFSURG01 DFSURG02 DFSURG03 DFSURG04 DFSURG05 DFSURG06 DFSURG07 DFSURG08 DFSURG09</td>
<td>DFSUIN01 DFSUIN02 DFSUIN03 DFSUIN04 DFSUIN05 DFSUIN06 DFSUIN07 DFSUIN08 DFSUIN09</td>
<td>All partitions are unloaded. All data sets unloaded must be input into the reload. Each file has a BMC header record; only the last file has a trailer record.</td>
</tr>
<tr>
<td></td>
<td>Five unload subtasks</td>
<td>Five reload subtasks</td>
<td>PARALLEL(YES) is required</td>
</tr>
<tr>
<td>PART(ALL) LIMITS(PART) PARALLEL(YES) is required</td>
<td>DFSURG01 DFSURG02 DFSURG03 DFSURG04 DFSURG05 DFSURG06 DFSURG07 DFSURG08 DFSURG09</td>
<td>DFSUINPT file1 file2 file3 file4 file5 file6 file7 file8 file9</td>
<td>All partitions are unloaded. All data sets unloaded must be input into the reload. Each file has a BMC header record; only the last file has a trailer record.</td>
</tr>
<tr>
<td></td>
<td>Five unload subtasks</td>
<td>One reload subtask</td>
<td></td>
</tr>
</tbody>
</table>
### Index Build function

The Index Build function is used to build indexes for IMS/ESA partitions and HALDBs.

#### IMS/ESA partition support

When a database is partitioned, an index is shared across all partitions. When you reorganize a database, index information is written to a WF1 file. If you reorganize some, but not all, partitions, a WF1 file is created but it will contain information only from the partitions that you reorganized.

### Table 217  IMS/ESA partition support product examples (part 2 of 2)

<table>
<thead>
<tr>
<th>Unload function keywords</th>
<th>Unload function DD statements</th>
<th>Reload function DD statements</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART(2,4,6) LIMITS(PART) PARALLEL(YES) is required</td>
<td>DFSURG02 DFSURG04 DFSURG06 Three unload subtasks</td>
<td>DFSUIN02 DFSUIN04 DFSUIN06 Three reload subtasks PARALLEL(YES) is required</td>
<td>All data sets that were unloaded must be reloaded. You cannot reload a subset of what was unloaded. The suffix of each file name must correspond to the partition number.</td>
</tr>
<tr>
<td>PART(2,4,6) LIMITS(PART) PARALLEL(YES) is required</td>
<td>DFSURG02 DFSURG04 DFSURG06 Three unload subtasks</td>
<td>DFSUINPT file1 file2 file3 One reload task</td>
<td>All HD Unload files must be concatenated in the correct order and used as input to the reload. You cannot reload a subset of what was unloaded.</td>
</tr>
<tr>
<td>PART(ALL) no LIMITS keyword</td>
<td>DFSURGU1 DFSURGU2 can only be a second copy of HD Unload file One unload subtask</td>
<td>DFSUINPT One reload subtask</td>
<td>All partitions are unloaded and reloaded using one file.</td>
</tr>
<tr>
<td>PART(2,7,9) no LIMITS keyword</td>
<td>DFSURGU1 DFSURGU2 can only be a second copy of HD Unload file One unload subtask</td>
<td>DFSUINPT One reload subtask</td>
<td>Partitions 2, 7, and 9 are unloaded and reloaded. One HD Unload file is created.</td>
</tr>
</tbody>
</table>
PART (database partitioning)

If you are using a WF1 file as input, the PART keyword is generally not valid. (For the exception, see the following tip.) The Reload function creates a header record in the WF1 file that indicates which partitions were loaded. When the Index Build function reads the WF1 file, it looks for the Reload function header record.

---

**TIP**

While it is generally not valid to build selected secondary indexes from a WF1 file, there is a special subset of DBD and RECON partition definitions that do support this function. Conceptually, selected WF1 secondary index build is supported when there is an exact match between every main database partition and every secondary index partition. Every main database partition must have one and only one secondary index partition that points into it for all secondary indexes. To use this function, code the main DNB part names in both the UNLOAD step and in the INDEX BUILD step. The load step must use ILDS(INIT) and can pick up the list of main DB partitions from the unload file header.

The index data set must be empty or defined as REUSE when you process all partitions. If it is not, the Index Build function terminates. The Index Build function assumes that you are processing all partitions in the following situation: If the Index Build function processes WF1 input and does not find the Reload function header record, it assumes that the WF1 file contains records for the entire database (all partitions).

The index data set must not be empty when you process selected partitions. If you try to process selected partitions and the index data set is empty, the Index Build function terminates. You can process selected partitions in the following ways:

- The Index Build function finds the Reload function header record indicating that selected partitions were loaded.
- You specified PART(partitionName, ...) or PART(nn,nn,nn).

Index records for the partitions that were not reorganized are retained, and the Index Build function merges the index data from the reorganized partitions with the data from the partitions that were not reorganized.

---

**WARNING**

If you unloaded selected partitions and created the DFSURWF1 file by using the Reload function, you must use the Index Build function to build the secondary indexes. Using the Index Build function is the only way to ensure that the secondary indexes contain all records for each partition.

The Index Build function issues a return code of 4 if any partition involved in the reorganization does not create records for a secondary index. The Index Build function continues processing.

If you specify PART(partitionName, ...) or PART(nn,nn), the IDCAMS keyword is ignored.
HALDBs

For information about building a subset of index partitions for a HALDB, see “HALDBs” on page 259.

Reorg function

To reorganize selected partitions, you must specify SHARE(YES).

Prefix Resolution/Update function

You must update and resolved prefixes for all database partitions at the same time.

Online/Defrag function

Only one Online/Defrag reorganization can execute against a partition at a time. Separate jobs can execute against different partitions simultaneously.

PCP

The PCP keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 218 describes the keyword.

Table 218 Invoke POINTER CHECKER PLUS keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activate the interface to the Hash Checking technique of POINTER CHECKER PLUS® to verify the pointers in the database while taking an image copy of the database.</td>
<td>PCP(Y)</td>
</tr>
<tr>
<td>Do not activate the interface to POINTER CHECKER PLUS.</td>
<td>PCP(N)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> PCP(N)</td>
<td></td>
</tr>
</tbody>
</table>

If you specify PCP(Y), the POINTER CHECKER PLUS load modules must be included in the STEPLIB concatenation in the JCL.

For more information, see “POINTER CHECKER PLUS interface” on page 334.
PDX (Unload, Reload, and Index Build functions)

The PDX keyword is valid with the BUILD, RELOAD, and REORG (Online Reorg function and Reorg function) commands. You can specify this option in a global options module or in PLUSIN. Table 219 describes the keyword.

Table 219  PDX keyword (Unload, Reload, and Index Build functions)

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continue if the PDX is unavailable, and end with return code nnnn.</td>
<td>PDX(OKAYnnnn)</td>
</tr>
<tr>
<td>Terminate if the PDX is unavailable, and end with abend/return code nnnn.</td>
<td>PDX(STOPnnnn)</td>
</tr>
</tbody>
</table>

The PDX keyword lets you specify the action the utility takes if the PDX is unavailable. You can continue with the job or terminate. Specify a 1- to 4-digit abend or return code in the range 0 through 4095.

You can set the default value for the PDX option by using the ISPF panels or the equivalent batch job. Sample JCL is in member #DBGGLBL of the sample library.

**NOTE**

PDX is not supported for HALDBs.

PDX (Image Copy function)

The PDX keyword is optional on the GLBL command. This keyword is available in ICPSYSIN; it is also available in the global options module. Table 220 describes the keyword.

Table 220  Image Copy function PDX data set name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the name of the PDX data set that should be used to retrieve</td>
<td>PDX(pdxName)</td>
</tr>
<tr>
<td>database specific options and to write processing statistics for the</td>
<td></td>
</tr>
<tr>
<td>Image Copy function.</td>
<td></td>
</tr>
<tr>
<td>Do not use a PDX data set for the Image Copy function.</td>
<td>PDX(NULLFILE)</td>
</tr>
<tr>
<td></td>
<td>PDX(DUMMY)</td>
</tr>
</tbody>
</table>

**NOTE**

PDX is not supported for HALDBs.
PDX (Prefix Resolution/Update function)

This keyword is used if the PDX DD statement is not included in the execution JCL. The value is the 1- to 44-character data set name.

For more information, see “Global options modules” on page 280 and “Monitoring and reporting options” on page 334.

PDX (Prefix Resolution/Update function)

The PDX keyword is valid with the Prefix Resolution/Update function. You can specify this option in a global options module or in PLUSIN. Table 221 describes the keyword.

Table 221 PDX keyword (Prefix Resolution/Update function)

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use pdxName to store statistical data.</td>
<td>PDX(pdxName)</td>
</tr>
<tr>
<td>Internal default: none</td>
<td></td>
</tr>
</tbody>
</table>

You can use a PDX to store statistical data that the Prefix Resolution/Update function captures while processing the data. For more information about the PDX, see the BMC IMS Database Supplemental Utilities Manual.

PDX (Online/Defrag function)

The PDX keyword is valid with the OREORG command. You can specify this option in the PLUSIN control statement or the $OLRPARM global options macro. Table 222 describes the keyword.

Table 222 PDX keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continue if the PDX is unavailable, and end with return code nnnn.</td>
<td>PDX(OKAYnnnn)</td>
</tr>
<tr>
<td>Terminate if the PDX is unavailable, and end with abend/return code nnnn.</td>
<td>PDX(STOPnnnn)</td>
</tr>
<tr>
<td>Internal default: OKAY0004</td>
<td></td>
</tr>
</tbody>
</table>

The PDX keyword lets you specify the action that Online/Defrag takes if PDXDSN is not specified. You can continue with the job or terminate. Specify a 1- to 4-digit abend or return code in the range 0 through 4095.
PDXPARMS

The PDXPARMS keyword is valid with the UNLOAD, RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. This keyword is available in PLUSIN; it is not available as a global option. Table 223 describes the keyword.

Table 223  PDX member name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use member as the PDX member name to retrieve options from the PDX.</td>
<td>PDXPARMS(member)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: Use the DBD name specified in the EXEC statement’s PARM operand.</td>
<td></td>
</tr>
</tbody>
</table>

The PDXPARMS keyword lets you specify the DBD name portion of the PDX key used to retrieve options. If the PDXPARMS member is not found in the PDX, the specification for the PDX keyword is followed (OKAY or STOP).

PRIMIND

The PRIMIND keyword is valid with the BUILD command. This keyword is available in PLUSIN; it is not available as a global option. Table 224 describes the keyword.

Table 224  Primary index keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the primary index along with all secondary indexes.</td>
<td>PRIMIND(YES)</td>
</tr>
<tr>
<td>Do not build the primary index when building the secondary indexes.</td>
<td>PRIMIND(NO)</td>
</tr>
<tr>
<td>Build only the primary index.</td>
<td>PRIMIND(ONLY)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: NO</td>
<td></td>
</tr>
</tbody>
</table>

Table 225 shows keywords related to the PRIMIND keyword.

Table 225  PRIMIND keyword and other keywords

<table>
<thead>
<tr>
<th>Keywords</th>
<th>Effect on related keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRIMIND(ONLY) and SECINDEX()</td>
<td>No indexes will be built, and the job will abend. If you specify PRIMIND(ONLY) and SECINDEX, the Index Build function cannot tell whether you want to build only the primary index or build the primary index as well as secondary indexes.</td>
</tr>
</tbody>
</table>
Index Build function

To build primary indexes, you must specify PRIMIND(ONLY). The following additional rules apply to building primary indexes:

- You cannot specify both the PART and SECINDEX parameter.
- The partition list must specify the list of primary database partitions.
- You cannot specify the following combinations:
  - ILDS(YES) and PRIMIND(ONLY)
  - ILDS(ONLY) and PRIMIND(YES)

The following JCL shows an example that results in primary index builds for partitions CUST2A and CUST3A.

```
BUILD DBD(CUSTOMER) PRIMIND(ONLY) PART(CUST2A,CUST3A)
```

PRTOLC

The PRTOLC keyword is valid with the REORG (Online Reorg function and Reorg function) command and the Prefix Resolution/Update function. You can specify this option in the environment setup member (DLIGSET0), a Prefix Resolution/Update function global options module, or in PLUSIN. Table 226 describes the keyword.

Table 226  Print logical child orphans keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print logical child orphans.</td>
<td>PRTOLC(Y)</td>
</tr>
<tr>
<td>Do not print logical child orphans.</td>
<td>PRTOLC(N)</td>
</tr>
<tr>
<td>Internal default: Print logical child orphans.</td>
<td></td>
</tr>
</tbody>
</table>

Logical child orphans are logical child records with no logical parent. Typically, this situation is not permitted; therefore, specify PRTOLC(Y).

PRTOLP

The PRTOLP keyword is valid with the REORG (Online Reorg function and Reorg function) command and the Prefix Resolution/Update function. You can specify this option in the environment setup member, a Prefix Resolution/Update function global options module or in PLUSIN. Table 227 on page 522 describes the keyword.
Logical parent orphans are logical parent records with no logical children. Typically, this is a common occurrence; however, if this situation is not allowed for your database, specify PRTOLP(Y) to help determine the problem.

The xxxPRINT keyword is valid with the BUILD, RELOAD, REORG (Online Reorg function and Reorg function), UNLOAD, and OREORG commands and with the HD Sort utility. You can specify this option for the BUILD, RELOAD, REORG, and UNLOAD commands in the environment setup member (DLIGSET0) or in PLUSIN. You can specify this option for the HDSORT command only in PLUSIN. Table 228 describes the keyword.

<table>
<thead>
<tr>
<th>Table 227 Print logical parent orphans keyword</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>To do this</strong></td>
</tr>
<tr>
<td>Print logical parent orphans.</td>
</tr>
<tr>
<td>Do not print logical parent orphans.</td>
</tr>
<tr>
<td>Internal default: Do not print logical parent orphans.</td>
</tr>
</tbody>
</table>

The xxxPRINT keyword specifies the name of the data set that contains reports generated by MAXM Reorg solutions. If you do not specify the xxxPRINT keyword, MAXM Reorg solutions dynamically allocate the BMCPRINT data set, based on MSGCLASS values specified on your JOB statement.
When you specify the xxxPRINT keyword, supply a corresponding ddname in your JCL. For example, if you specify FRFPRINT(FRFREPTS), include a FRFRPTS DD statement in the JCL. If you do not specify a corresponding DD statement, a SYSOUT=* output file is dynamically allocated with the name specified with the keyword.

**PSB**

The PSB keyword is valid with the OREORG command. This keyword is available in PLUSIN; it is not available as a global option. Table 229 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the PSB to be used for the BMP.</td>
<td>PSB(psbName)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> none</td>
<td></td>
</tr>
</tbody>
</table>

**Online/Defrag function**

The PSB keyword is required.

When you execute Online/Defrag, the Online/Defrag job uses a BMP to coordinate the reorganization. The PSB keyword names the PSB to be used for the BMP subtask. Online/Defrag validates the PSB and ensures that it is sensitive to all segments of the database and complies with the required Online/Defrag structure.

The PSB must be defined in the IMS definition used by the online control region in which the Online/Defrag job runs. Two PCBs are required for each DBD. Both PCBs must be sensitive to all segments, and both PCBs must be named. PROCOPT=A is required. The PSBs must have the structure shown in Figure 88.

**Figure 88 Required PSB structure**

```
PCB TYPE=DB,NAM=dbname,PROCOPT=A,PCBNAME=anyname1,LIST=YES
SENSEG NAME=rootseg
SENSEG NAME=depsegl
.
.(all segments)
.
PCB TYPE=DB,NAM=dbname,PROCOPT=A,PCBNAME=anyname2,LIST=YES
SENSEG NAME=rootseg
SENSEG NAME=depsegl
.
.(all segments)
.
PSBGEN PSBNAME=psbname,LANG=COBOL,CMPAT=Y
END
```
The PTRCHECK keyword is valid with the UNLOAD command. You can specify this option in an Unload function global options module or in PLUSIN. Table 230 describes the keyword.

**Table 230  Check physical pointers keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check physical pointers.</td>
<td>PTRCHECK(PHY)</td>
</tr>
<tr>
<td>Do not check pointers.</td>
<td>PTRCHECK(NO)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: NO</td>
<td></td>
</tr>
</tbody>
</table>

The Unload function uses the PTF and PCF (or HF) pointers to progress through a database record and automatically checks them for validity. Use the PTRCHECK(PHY) keyword to check the PTB, PCL, and PP pointers.

If you specify PTRCHECK(PHY) and the Unload function detects an invalid pointer, it takes a snap dump of the bad block and reports the actual and expected pointer values. The PTRERROR keyword controls whether the Unload function continues after detecting a bad pointer. For more information, see “PTRERROR” on page 524.

You can set the default value for the PTRCHECK option by using the Unload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

**PTRERROR**

The PTRERROR keyword is valid with the DBSCAN and UNLOAD commands. You can specify this option in a global options module or in PLUSIN. Table 231 describes the keyword.

**Table 231  Pointer error keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continue after pointer errors (no limit).</td>
<td>PTRERROR(ACCEPT)</td>
</tr>
<tr>
<td>Continue after pointer errors (up to (n) errors).</td>
<td>PTRERROR(ACCEPT,(n))</td>
</tr>
<tr>
<td>Abend after a pointer error.</td>
<td>PTRERROR(ABEND)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: ABEND</td>
<td></td>
</tr>
</tbody>
</table>
The Unload function uses the PCF and PTF (or HF) pointers to move in hierarchical sequence through a database record. If you specify PTRERROR(ABEND) and any of these pointers are bad, the unload terminates. You can use the PTRERROR keyword to bypass these errors. The Unload function can skip the bad segment or a portion of the bad record and continue.

When the Unload function encounters an error, it snaps the block containing the bad database pointer and generates a BMC message. When the unload is finished, you can use this information to fix the problem. For more information, see “Bad Block/Pointer report” on page 585.

If you specify PTRERROR(ACCEPT,nnnnn), the Unload function continues until it encounters nnnnn pointer errors. The value for nnnnn can be any number from 1 through 32,767. If you use the default PTRERROR(ABEND), the Unload function terminates when it detects the first pointer error.

---

**NOTE**

If you accept pointer errors and pointer errors exist, the Unload function job step issues return code 8 and continues.

---

You can set the default value for the PTRERROR option by using the Unload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

---

### REORGMSG

The REORGMSG keyword is valid with the OREORG command. You can specify this option in PLUSIN. Table 232 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display detailed information about each record</td>
<td>REORGMSG(value)</td>
</tr>
<tr>
<td>that was reorganized.</td>
<td></td>
</tr>
</tbody>
</table>

*Internal default: NO*

The REORGMSG keyword is valid with MODE(SCAN). Valid values are YES and NO.

---

**WARNING**

Use the keyword with caution because the messages might be voluminous.
The REPORTS keyword is valid with the UNLOAD, RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. You can specify these options in a Reload function or Unload function global options module or in PLUSIN. Table 233 describes the keyword.

**Table 233  Reports keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate the Segment I/O Requirements Matrix report (Unload function and Reload function).</td>
<td>REPORTS(IOSTATS(YES))</td>
</tr>
<tr>
<td>List the root keys of the nnn largest records on the Largest Database Records report (Unload function and Reload function).</td>
<td>REPORTS(KEYS(nnn))</td>
</tr>
<tr>
<td>Generate the Distribution of HDAM RAP Chain Lengths Report (Reload function).</td>
<td>REPORTS(RAPCHAIN(YES))</td>
</tr>
<tr>
<td>Generate the Distribution of Database Record Sizes report with default ranges (Unload function and Reload function).</td>
<td>REPORTS(RCDDIST(YES))</td>
</tr>
<tr>
<td>Generate the Distribution of Database Record Sizes report with user-specified size ranges (Unload function and Reload function).</td>
<td>REPORTS(RCDDIST(n1,n2,...))</td>
</tr>
<tr>
<td>Specify the number of blocks to scan for the HDAM Root Placement Relative to Its Home Block report (Unload function).</td>
<td>REPORTS(ROOTDIST(-nn+mm))</td>
</tr>
</tbody>
</table>

*Internal default:* See Table 234.

The REPORTS keyword and its subkeywords enable you to manipulate some Unload function and Reload function reports. Table 234 describes the REPORTS subkeywords.

**Table 234  REPORTS subkeywords (part 1 of 2)**

<table>
<thead>
<tr>
<th>Subkeyword</th>
<th>Accepted values</th>
<th>Default values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOSTATS</td>
<td>(NO) (YES)</td>
<td>RELOAD: (YES)</td>
<td>Generate the Segment I/O Requirements Matrix report.</td>
</tr>
<tr>
<td>KEYS</td>
<td>(nnn) max 999</td>
<td>(10)</td>
<td>Set number of records for Database Record Statistics report.</td>
</tr>
<tr>
<td>RAPCHAIN</td>
<td>(NO) (YES)</td>
<td>(NO)</td>
<td>Generate the Distribution of HDAM RAP Chain Lengths report.</td>
</tr>
</tbody>
</table>
You can set the default values for the REPORTS option by using the Reload function or Unload function ISPF panels or the equivalent batch jobs. Sample JCL is in member #DBUGLBL of the sample library.

Use the subkeywords as follows:

**IOSTATS**

Use the IOSTATS subkeyword to generate the Segment I/O Requirements Matrix report. This matrix shows the number of I/Os required by DL/I to go from any segment type to another segment type. “Segment I/O Requirements Matrix” on page 612 describes the report.

**KEYS**

Use the KEYS subkeyword to generate the Largest Database Records report. This report lists the root keys and sizes of the \( n \) largest database records. You can specify \( n \) as 0 to 999. If you specify 0, the report will not be generated. “Largest Database Records” on page 600 describes the report.

**RAPCHAIN**

Use the RAPCHAIN subkeyword to generate the Distribution of HDAM RAP Chain Lengths report. The report shows the root synonym chain length and the number of chains. “Distribution of HDAM RAP Chain Lengths” on page 595 describes the report.

**NOTE**

Reload function performance can be degraded when generating the Distribution of HDAM RAP Chain Lengths report.
RCDDIST

Use the RCDDIST subkeyword to generate the Distribution of Database Record Sizes report and specify the different ranges the utility will use when producing this report. Each value you specify sets the next upper limit (in bytes). The utilities automatically handle any database records that exceed the largest limit you specify. You can specify a maximum of nineteen 7-byte entries. The default is to generate the report with ranges by hundreds (100, 200,...) up to 1000, then by thousands (1000, 2000,...) up to 10,000. “Distribution of Database Record Sizes” on page 594 describes the report.

ROOTDIST

Use the ROOTDIST subkeyword to generate the HDAM Root Placement Relative to its Home Block report. This report shows where the HDAM or PHDAM root segments were found relative to their home blocks. Use the format ROOTDIST(-nn+mm), where nn is the number of blocks ahead of the home block and mm is the number of blocks following the home block. The value of nn and mm can be 0 to 99.

For example, ROOTDIST(-10+5) requests a distribution of roots found in the home block, the 10 adjacent previous blocks, and the 5 adjacent subsequent blocks. If roots are found beyond the nn or mm values, they are accumulated in the nn or mm buckets.

If you do not want to generate the HDAM Root Placement Relative to Its Home Block report, specify 0 for both scanning ranges. The default is to not generate the report.

“HDAM Root Placement Relative to Its Home Block” on page 598 describes the report.

RESUME

The RESUME keyword is valid with the OREORG command. You can specify this option in PLUSIN. Table 235 describes the keyword.

Table 235  RESUME keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in HSRSYSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify that the reorganization is not a continuation.</td>
<td>RESUME(NO)</td>
</tr>
<tr>
<td>Specify that the reorganization is a continuation.</td>
<td>RESUME(YES)</td>
</tr>
<tr>
<td>Specify that the reorganization is a continuation, and specify the key value of the restart record.</td>
<td>RESUME(YES, keyValue)</td>
</tr>
</tbody>
</table>

Internal default: (NO)
Use the RESUME keyword to specify whether the reorganization is a continuation of a prior reorganization that was prematurely terminated.

If RESUME(YES) is specified, Online/Defrag resumes the reorganization at the restart record identified in the PDX data set. If the restart record cannot be retrieved, the reorganization starts anew.

If RESUME(YES,...) is specified, you must specify the key value of the record with which to continue the reorganization. If no key value is specified, the information in the PDX restart record (if present) is used. If the restart record cannot be retrieved, the reorganization starts anew.

If you want to interrupt and later resume the same reorganization job, BMC recommends that you use the /F jobName, SHUTDOWN command to interrupt the reorganization. However, you can resume a reorganization after any method of termination (excluding MVS system failure).

Upon shutdown, Online/Defrag writes the restart record to the PDX.

For more information about the PDX data set name, see “PDX (Online/Defrag function)” on page 519.

### RMBYTES

The RMBYTES keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. You can specify this option in the environment setup member (DLIGSET0) or in PLUSIN. Table 236 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the segment’s VL SIZE field to accumulate database record bytes for segments smaller than the specified MINBYTES value.</td>
<td>RMBYTES(SEGSIZE)</td>
</tr>
<tr>
<td>Use the segment’s MINBYTES value to accumulate database record bytes.</td>
<td>RMBYTES(MINBYTES)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> MINBYTES</td>
<td></td>
</tr>
</tbody>
</table>

The RMBYTES keyword specifies how the Reload function calculates the database record size to determine when to place consecutively inserted segments into overflow. When the value contained in the VL SIZE field of a variable-length segment is less than the value specified for its MINBYTES parameter, the RMNAME operand’s maximum insert bytes value has been exceeded.
Specify MINBYTES or SEGSIZE according to your requirements:

- To use the MINBYTES value, specify \texttt{RMBYTES(MINBYTES)}. The value specified for the segment’s MINBYTES parameter is accumulated. This is the method IMS uses, and it is the default for the Reload function.

- To use the segment’s actual length (VL SIZE field), specify \texttt{RMBYTES(SEGSIZE)}. The value in the VL SIZE field (first two bytes of data) is accumulated. Reload function uses the \texttt{RMBYTES(SEGSIZE)} method.

The \texttt{RMBYTES} keyword applies only to HDAM and PHDAM databases that meet the following criteria:

- The maximum insert bytes parameter of the RMNAME operand is specified in the DBD; for example, \texttt{RMNAME=(DFSHDC40,5,2000,1000)} where the maximum insert bytes parameter is 1000.

- The DBD contains one or more segment statements with the MINBYTES parameter of the \texttt{BYTES=} operand specified (variable-length segments).

\textbf{NOTE}

The \texttt{RMBYTES} keyword does not affect the actual (MINBYTES) segment size stored in the RAA.

\section*{ROUTCDE (Image Copy function)}

The ROUTCDE keyword is optional on the GLBL, COPY, and REORG commands. This keyword is available in ICPSYSIN; it is also available in the global options module. Table 237 describes the keyword.

\begin{table}[h]
\centering
\begin{tabular}{|l|l|}
\hline
To do this & Specify this in ICPSYSSIN \\
\hline
Specify a list of routing code numbers (1–15) that indicates which types of MVS operator consoles are to receive Image Copy function monitoring messages. & ROUTCDE\texttt{(nn,nn,...)} \\
\hline
\textbf{Internal default:} ROUTCDE\texttt{(1,7,11)} & \\
\hline
\end{tabular}
\caption{Image copy monitor routing code keyword}
\end{table}

You can specify a string of up to 20 characters. Separate the numbers with commas. Each comma counts as one character.

For more information, see “Monitoring and reporting options” on page 334.
The SCAN keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. You can specify this option in a Reload function global options module or in PLUSIN. Table 238 describes the keyword.

### Table 238  HIDAM and PHIDAM space search keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search for space in a previous block (limit number of blocks to (nn)).</td>
<td>SCAN((nn))</td>
</tr>
<tr>
<td>Do not search for space in a previous block.</td>
<td>SCAN(0)</td>
</tr>
<tr>
<td>Internal default: 0</td>
<td></td>
</tr>
</tbody>
</table>

You can force the Reload function to search for space in a previous block before starting a new block for non-root segments in HIDAM and PHIDAM databases. This keyword makes the Reload function space search algorithm similar to the IMS algorithm. The Reload function allows you to specify a limit on the number of blocks to search.

Searching for space in a previous block has several implications. The advantage is that the DASD space is packed much tighter. The amount of DASD space required to hold a given database could be less. The disadvantage is that the number of I/Os required to retrieve all segments of a given database record might increase. Performance might be adversely affected when you are using the database, particularly if you have long twin chains of relatively small segments.

**NOTE**
The search for space is applicable to non-root segments only. If a root segment does not fit in the current block, it is placed in the next block.

You can set the default value for the SCAN option by using the Reload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

---

The SCAN keyword is valid with the RELOAD and REORG (Online Reorg function and Reorg function) commands. You can specify this option in a Reload function global options module or in PLUSIN. Table 239 on page 532 describes the keyword.
The Reload function allows you to control the placement of all HDAM or PHDAM segments that do not fit in the RAA most favored block, the randomized block. The Reload function can scan the specified number of adjacent blocks for free space before putting the segment into the overflow area.

When you specify SCAN(NO), the Reload function and the IMS techniques of searching for free space are skipped. Any segments unable to fit in the home block are placed in the overflow area.

The number of blocks you specify affects the region size needed to run the Reload function. If the number of blocks is large or the blocks themselves are large, increase the region size by adding the product of the number of blocks scanned, multiplied by the number of bytes per block. If you use the Reload function space search method, you can also control the manner in which the search for space proceeds.

The flip-flop technique (shown in Figure 89 on page 533) scans the block immediately behind the prime block, then the block immediately ahead. Next it scans the block two blocks behind, then the block two ahead (and so on), until it finds space or reaches the user-specified number of blocks behind or ahead of the prime block. “SEARCH n” in a block shows the block is being searched for space to insert the segment.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use bidirected method of the Reload function space search and search backward first.</td>
<td>SCAN(-bbbbb)</td>
</tr>
<tr>
<td>Use bidirected method of space search and search forward first.</td>
<td>SCAN(+b)</td>
</tr>
<tr>
<td>Bypass space search and write the segment to overflow.</td>
<td>SCAN(0)</td>
</tr>
<tr>
<td>Use flip-flop method of space search and search backward first.</td>
<td>SCAN(-n)</td>
</tr>
<tr>
<td>Use flip-flop method of space search and search forward first.</td>
<td>SCAN(+n)</td>
</tr>
</tbody>
</table>

**Internal default**: -10+10
The bidirected technique (shown in Figure 90) scans a user-specified number of blocks in one direction, then a user-specified number of blocks in the other direction. The starting direction (forward or backward) is also user-specified. “SEARCH n” in a block indicates the block is being searched for space to insert the segment.

You can set the default value for the SCAN option by using the Reload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.
SECINDEX

The SECINDEX keyword is valid with the BUILD and COPY commands. This keyword is available in PLUSIN; it is not available as a global option. Table 240 describes the keyword.

Table 240  Secondary index name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build selected secondary indexes.</td>
<td>SECINDEX(secIndex1,secIndex2,...)</td>
</tr>
<tr>
<td>Build all secondary indexes.</td>
<td>SECINDEX()</td>
</tr>
<tr>
<td>Build selected HALDB secondary index partitions</td>
<td>SECINDEX(secIndexPart1,secIndexPart2,...)</td>
</tr>
<tr>
<td>Copy the DBD and all secondary indexes.</td>
<td>SECINDEX(YES)</td>
</tr>
<tr>
<td>Copy only the DBD.</td>
<td>SECINDEX(NO)</td>
</tr>
<tr>
<td>Copy only the secondary indexes of the DBD.</td>
<td>SECINDEX(ONLY)</td>
</tr>
</tbody>
</table>

Internal default:
- For the BUILD Command, the default is SECINDEX().
- For the COPY Command, the default is SECINDEX(NO).

BUILD command

The SECINDEX keyword enables you to specify as many as 30 secondary indexes to be built. All secondary indexes must be from the primary databases that were specified in the DBD keyword.

If you do not specify any secondary indexes, the Index Build function builds all secondary indexes for the databases. To build all secondary indexes, specify SECINDEX() or do not include the SECINDEX keyword with the BUILD command.

The SECINDEX keyword also enables you to specify up to 30 HALDB secondary index partitions to be built. All partitions must be from one or more secondary indexes from the primary database that was specified in the DBD keyword. For more information about building a subset of index partitions for a HALDB, see "HALDBs" on page 259.

You cannot specify both the SECINDEX and PART keywords.

COPY command

Dynamic allocation is required when SECINDEX is specified with the Copy function.
SEGPOS

The SEGPOS keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. This keyword is available in PLUSIN; it is not available as a global option. Table 241 describes the keyword.

Table 241  Segment position keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Place the specified segments near the root.</td>
<td>SEGPOS(segName,n, segName,n,...segName,n)</td>
</tr>
</tbody>
</table>

Internal default: none

When you reload a database, you can use the SEGPOS keyword to specify which segments to place next to the root. When you place the segments you access frequently next to the root, you can improve performance.

In a traditional reorganization, a database is reloaded in hierarchical sequence. This could cause certain segments that are low in the hierarchy to never be near the root. When you use the SEGPOS keyword, the segments you specify with the keyword are placed near the root rather than low in the hierarchy.

You can place multiple segments next to the root. The Reload function places the segments next to the root on a one-by-one basis. When you specify a segment, you must also specify the number of segments of that type that you want near the root. Use the following syntax:

\[(\text{segname},n,\text{segname},n,...\text{segname},n)\]

For example, if you specified that you wanted three occurrences of SEGA and three occurrences of SEGB next to the root (SEGA,3,SEGB,3), the Reload function would sequence the segment types as follows: SEGA1, SEGB1, SEGA2, SEGB2, SEGA3, SEGB3. You can specify as many as 25 segments.

HDAM and PHDAM BYTE parameters (maximum bytes and bytes limit) are honored. When the block that contains the root fills up, the next segment type will be placed according to the SCAN keyword, then into the overflow area.

**NOTE**

The use of the SEGPOS keyword will increase CPU time.

If you specify the OVFLONLY keyword and the SEGPOS keyword in the same job step, the SEGPOS keyword is ignored.
SEGPOSO

The SEGPOSO keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. This keyword is available in PLUSIN; it is not available as a global option. The SEGPOSO keyword is an extension of the SEGPOS keyword and is used in conjunction with SEGPOS. The SEGPOSO keyword is only valid with HDAM databases. Table 242 describes the keyword.

Table 242  Segment position extension keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Place all segments in the overflow area, if it exceeds the limit specified in the SEGPOS specification.</td>
<td>SEGPOSO(ALL)</td>
</tr>
<tr>
<td>Place the specified segments in the overflow area, if it exceeds the limit specified in the SEGPOS specification. Maximum of 25 names.</td>
<td>SEGPOSO(segName1, segName2, ...)</td>
</tr>
</tbody>
</table>

Internal default: none

The following rules apply when using the SEGPOSO keyword:

- When the segment placement limit is reached, the specified segments are placed in the overflow area.
- If the SEGPOS keyword is not specified, the SEGPOSO keyword is ignored.
- If segments are named in the SEGPOSO specification, but are not named in the SEGPOS specification, the segments are ignored.

SEGSEL

The SEGSEL keyword is valid with the UNLOAD command. This keyword is available in PLUSIN; it is not available as a global option. Table 243 describes the keyword.

Table 243  Segment selection keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unload only the selected segment names.</td>
<td>SEGSEL(seg1,seg2,...)</td>
</tr>
</tbody>
</table>

Internal default: none

Use the SEGSEL keyword to unload only selected segment names. You can specify as many as 30 segment names. You cannot select virtual logical children segments.
When you specify the SEGSEL keyword, the data is written in hierarchical order, not the order listed in the SEGSEL keyword.

**SEQERROR**

The SEQERROR keyword is valid with the RELOAD and UNLOAD commands. You can specify this option in a Reload function or Unload function global options module or in PLUSIN. Table 244 describes the keyword.

**Table 244  Sequence error keyword**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continue after twin segment sequence errors (no limit) and place segment in unload or reload file. (Unload function and Reload function)</td>
<td>SEQERROR(ACCEPT)</td>
</tr>
<tr>
<td>Continue after twin segment sequence errors and place segment in unload or reload file. (Unload function and Reload function)</td>
<td>SEQERROR(ACCEPT,nnnnnn)</td>
</tr>
<tr>
<td>Abend after a twin segment sequence error. (Unload function and Reload function)</td>
<td>SEQERROR(ABEND)</td>
</tr>
<tr>
<td>Continue after twin segment sequence errors (no limit) and place segment in the SEQERROR data set. (Reload function)</td>
<td>SEQERROR(SAVE)</td>
</tr>
<tr>
<td>Continue after twin segment sequence errors and place segments (and their dependents) in the SEQERROR data set. (Reload function)</td>
<td>SEQERROR(SAVE,nnnnnn)</td>
</tr>
</tbody>
</table>

**Internal default**: ABEND

The IMS HD Unload utility does not check twin segments for sequence, but the IMS HD Reload utility does; therefore, it is possible to encounter a sequence error during the reload process. This sequence error could cause the IMS HD Reload utility to abend. Because it is much better to catch sequence errors during the unload process rather than during the reload process, the Unload function checks for sequence errors.

The Unload function always sequence-checks twin chains. If the Unload function detects a sequence error, it terminates by default. The SEQERROR keyword allows you to specify that the Unload function continue with the unload. When the unload has ended, the Unload function gives you the appropriate data to correct the sequence error.

**NOTE**

If you accept sequence errors and sequence errors exist, the Unload function job step issues return code 8 and continues.
The Reload function also allows you to continue after a sequence error. You can specify a limit on the number of sequence errors that occur before the Reload function abends. If you decide to continue with the reload, the Reload function provides the appropriate data to correct the sequence error.

**NOTE**

You can use the Reload function SEQERROR(SAVE) keyword with the Unload function SEQERROR(ACCEPT) keyword to move twin segment sequence errors to the SEQERROR data set, so you can load a clean database.

The Unload function takes the following actions when it encounters a sequence error:

- writes message BMC90213 to the BMCM SG data set
- determines whether to abend or continue, as shown in Table 245

**Table 245  SEQERROR and the Unload function**

<table>
<thead>
<tr>
<th>If you specify</th>
<th>Unload function</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEQERROR(ABEND)</td>
<td>Terminates with a 4094 abend</td>
</tr>
<tr>
<td>SEQERROR(ACCEPT)</td>
<td>Continues with the unload process, placing the out-of-sequence dependent segments in the HD Unload data set</td>
</tr>
<tr>
<td>SEQERROR(ACCEPT,nnnnnn)</td>
<td>Continues until nnnnn sequence errors are encountered, at which time it abends</td>
</tr>
</tbody>
</table>

The Reload function takes the following actions when it encounters a sequence error:

- writes messages BMC90104 and BMC90109 to the BMCM SG data set
- determines whether to abend or continue, as shown in Table 246

**Table 246  SEQERROR and the Reload function**

<table>
<thead>
<tr>
<th>If you specify</th>
<th>Reload function</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEQERROR(ABEND)</td>
<td>Terminates with a 4094 user abend</td>
</tr>
<tr>
<td>SEQERROR(ACCEPT)</td>
<td>Performs one of the actions listed in Table 247, depending on the type of database and the type of segment out of sequence</td>
</tr>
<tr>
<td>SEQERROR(SAVE)</td>
<td>Places segment and all segments hierarchically dependent on it in the SEQERROR data set and reloads a clean database</td>
</tr>
</tbody>
</table>

**NOTE**

If you have SEQERROR(ACCEPT) or SEQERROR(ACCEPT,nm) under the load API that it is dynamically changed to SEQERROR(ABEND).

**Table 247 on page 539** shows error handling for twin segment sequences.
Segments written to the SEQERROR data set are available for review and re-insertion by a user-written program. An example of an assembler program that processes the SEQERROR data set is located in the member SEQERASM of the BMC sample library.

**NOTE**

The use of the SEQERROR(ACCEPT) or SEQERROR(SAVE) keywords in the Reload function will increase CPU time.

You can set the default value for the SEQERROR option by using the Reload function or Unload function ISPF panels or the equivalent batch jobs. Sample JCL is in member #DBGUBLBL of the sample library.

### SEQERROR data set

The Reload function can allocate the SEQERROR data set if you specify SEQERROR(SAVE). You can tell the Reload function how to allocate the SEQERROR data set by specifying the following options through the ISPF interface:

- **Y** for Continue After a Twin Segment Sequence Error
- **Y** for Save sequence errors
- Specify values for Volume Serial (optional), Unitname, Primary Cylinders, Secondary Cylinders, and High Level Qualifier on the Allocation Parameters panel.

You can specify the SEQERROR DD statement in your JCL. If you specify the SEQERROR DD statement, the allocation parameters specified in the PDX are ignored. Because the Reload function always dynamically allocates the data set to DASD, you must specify a DD statement if you want to direct the data set to a tape device.

---

### Table 247 Twin segment sequence error handling

<table>
<thead>
<tr>
<th>Database type</th>
<th>Segment types</th>
<th>SEQERROR keyword</th>
<th>Reload function action</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>all</td>
<td>SAVE</td>
<td>Place segment and all segments hierarchically dependent on it in the SEQERROR data set and reload a clean database.</td>
</tr>
<tr>
<td>HIDAM, PHIDAM, or HISAM</td>
<td>non-root</td>
<td>ACCEPT</td>
<td>Place segment in database (still out of sequence) and reload.</td>
</tr>
<tr>
<td></td>
<td>root</td>
<td>ACCEPT</td>
<td>Abnormally terminate.</td>
</tr>
<tr>
<td>HDAM and PHDAM</td>
<td>non-root</td>
<td>ACCEPT</td>
<td>Place segment in database (still out of sequence) and reload.</td>
</tr>
<tr>
<td></td>
<td>unique root (duplicate keys encountered)</td>
<td>ACCEPT</td>
<td>Chain segment off appropriate RAP and continue.</td>
</tr>
</tbody>
</table>
If required, the SEQERROR data set is allocated as follows:

```
DSN=dataSetName.dbdName.SEQERROR.yymmdd.TThhmms and
DISP=(NEW,CATLG)
```

The variables indicate the following information:

- `dataSetName` is the high-level data set name qualifier you specified.
- `dbdName` is the name of the database being loaded.
- `yymmdd` is the file creation date.
- `hhmmss` is the file creation time.

**SHARE**

The SHARE keyword is valid with the REORG (Online Reorg function and Reorg function) command. You can specify this option in a Reorg function global options module or in PLUSIN. Table 248 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable users to have read access to the database during the reorganization (Reorg function).</td>
<td>SHARE(YES) or SHARE(YES,READ)</td>
</tr>
<tr>
<td>Give the Reorg function exclusive control of the database during the reorganization.</td>
<td>SHARE(NO)</td>
</tr>
<tr>
<td>Enable users to have update access to the database during the reorganization (Online Reorg function).</td>
<td>SHARE(YES,UPDATE)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> NO</td>
<td></td>
</tr>
</tbody>
</table>

The SHARE keyword specifies whether users can have *read* or *update* access to the database during reorganization.

**WARNING**

Regardless of the value of the SHARE keyword, the Online Reorg function and the Reorg function always require *exclusive* control during the data set name swapping process. If you attempt a multiple step JCL allocation, exclusive control is lost and the database becomes unavailable.

DBRC must be active and the databases must be registered for SHARE(YES) to be valid.
In the Reorg function, if you want to reorganize only selected partitions of a partitioned database, you must specify SHARE(YES). In the Online Reorg function, you cannot reorganize selected partitions of a partitioned database. For more information, see “PART (database partitioning)” on page 508.

**Online Reorg function**

The Online Reorg function allows *update* access to the database during the reorganization when you specify SHARE(YES,UPDATE).

**Reorg function**

The Reorg function allows *read* (but not *update*) access to the database during the reorganization. If you specify SHARE(YES), the Reorg function allows sharing by issuing authorization requests to DBRC for *read exclusive* access to the database data sets. This allows other users to read the database but ensures that no updates take place.

**NOTE**

If any output data sets have names that are registered to DBRC, SHARE(YES) is considered an error and the job step terminates.

If you specify SHARE(NO), the Reorg function obtains *exclusive* control during the entire reorganization process. The database is unavailable to other users.

You can set the default value for the SHARE option by changing the FRF$GLBL global options module. Sample JCL is in member #FRFGLBL of the sample library.

**SIUDSN**

The SIUDSN keyword is optional on the GLBL command. This keyword is available in ICPSYSIN; it is also available in the global options module. Table 249 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify a dummy data set name (1–44 characters) to use when notifying DBRC that a virtual image copy has been created.</td>
<td>SIUDSN(dataSetName)</td>
</tr>
</tbody>
</table>

Internal default: SIUDSN(BMC.SIUDUMMY.DSN)
A virtual image copy is not a real image copy; it is a dummy data set name recorded in DBRC as the image copy name of an index database. The dummy data set name must be cataloged.

The image copies defined to DBRC must be assigned the NOREUS parameter.

For more information, see “Virtual image copies” on page 330.

**SIUSORT**

The SIUSORT keyword is valid with the BUILD and REORG (Online Reorg function and Reorg function) commands. You can specify this keyword in the environment setup member (DLIGSET0) or in PLUSIN. Table 250 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sort and load index records internally (in the same address space).</td>
<td>SIUSORT(I)</td>
</tr>
<tr>
<td>Sort and load index records externally (in separate address spaces).</td>
<td>SIUSORT(E)</td>
</tr>
</tbody>
</table>

**Internal default:** (I)

The SIUSORT keyword enables you to sort and load indexes in separate address spaces. If you use separate address spaces to sort and load indexes, elapsed time for large indexes will be shorter. When you specify SIUSORT(E), the Index Build function creates one address space for each index.

When an SIUSORT(E) job is running, you will see a separate MVS job for each index. When all of the indexes are sorted and loaded, the output from the separate MVS jobs is merged into the original job.

**NOTE**

The installation defaults for your sort product might affect how the sort process works. To override the sort product’s options, add the appropriate DD statements for your sort product to the BMC job step. When you override the sort product’s options, the Index Build function propagates the DD statements to the separate address spaces. For DFSORT, add a DFSPARM DD statement; for SyncSort, add a $ORTPARM DD statement.
The Index Build function uses the MVS proc IEESYSAS when it creates the new address spaces. The job name given to the new address spaces is in the following format:

\[ Jnnnnxx \]

The variables indicate the following information:

- \( nnnn \) is the JES job number for the original job.
- \( xx \) is the sequential number used to make the job name unique.

**NOTE**

If ICP(Y) and secondary indexes are copied, SIUSORT(E) is overridden with SIUSORT(I).

---

**SNAPSHOT**

The SNAPSHOT keyword is valid with the REORG (Online Reorg function) and UNLOAD commands. This keyword is available in PLUSIN; it is not available as a global option. SNAPSHOT is not valid with the USERREAD command, and SNAPSHOT is not valid with the REORG command when using the GROUP command. Table 251 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Take a snapshot copy of the database being unloaded.</td>
<td>SNAPSHOT(Y)</td>
</tr>
<tr>
<td>Do not take a snapshot copy of the database being unloaded.</td>
<td>SNAPSHOT(N)</td>
</tr>
</tbody>
</table>

**Internal default: (N)**

If you have EXTENDED BUFFER MANAGER for IMS (XBM for IMS) or the SNAPSHOT UPGRADE FEATURE (SUF for IMS) installed and you specify SNAPSHOT(Y), XBM Snapshot provides a point-in-time image of the database to be used while subsequent updates are in progress.

The XBMSSID keyword is required. For more information, see “XBMSSID” on page 574.

For more information, see the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE User Guide and the EXTENDED BUFFER MANAGER and SNAPSHOT UPGRADE FEATURE Installation Guide.
Snapshot Copy is used for the reorganization. This eliminates a full copy of the database, as well as an unload to temporary data sets, thus saving DASD and the overhead associated with taking a copy. It also reduces the elapsed time of the reorganization.

**Unload function**

You can take a Snapshot Copy of a database while running the Unload function in stand-alone mode, but the HD Unload file that is created will not be valid for a reload if updates occurred during or following the unload.

**SORTIDX**

The SORTIDX keyword is valid with the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or in PLUSIN. Table 252 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run the Secondary Index Resolution function during Parent-Child Resolution.</td>
<td>SORTIDX(Y)</td>
</tr>
<tr>
<td>Do not run the Secondary Index Resolution function during Parent-Child Resolution.</td>
<td>SORTIDX(N)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: Do not run the Secondary Index Resolution function during Parent-Child Resolution.</td>
<td></td>
</tr>
</tbody>
</table>

You can run the Secondary Index Resolution function during Parent-Child Resolution (DFSURG10 or PRPURG20). Specify SORTIDX(Y) to have the secondary index records sorted. Specify SORTIDX(N) if you do not want the records sorted. If you specify SORTIDX(N), you must run the DFSURIDX data set through the Secondary Index Resolution function, the Index Build function, or the IMS Prefix Resolution utility.

**SORTIN**

The SORTIN keyword is valid with the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or in PLUSIN. Table 253 on page 545 describes the keyword.
The Prefix Resolution/Update function assumes that IMS-format work file records will be used as input to PRPURG10, PRPURG20, and PRPURG40 using the DFSURWF1 DD statement, and that BMC-format work file records will be used as input to these phases using the SORTIN DD statement. If you create only IMS-format records and want to read the IMS-format work file records on SORTIN, specify SORTIN(Y).

Specifying SORTIN(Y) will not improve performance because the IMS-format work file records must still be read using an input exit routine. The SORTIN option is provided for full compatibility with existing JCL. For best performance, create BMC-format records during database load processing and use them throughout the reorganization process.

### SORTWORK

The SORTWORK keyword is valid with the BUILD, RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands and with the HD Sort utility. You can specify the number of SORTWORKnn data sets, the number of primary cylinders, and the unit name for this option in the environment setup member (DLIGSET0). You can specify all parameters in PLUSIN. Table 254 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamically allocate sort work data sets.</td>
<td>SORTWORK(n,ccc,unitName, dataclas,mgmtclas,storclas)</td>
</tr>
<tr>
<td>Internal default: (3,100,SYSDA)</td>
<td></td>
</tr>
</tbody>
</table>

The SORTWORK keyword enables you to set parameters for dynamically allocated sort work data sets. If you do not specify the SORTWORK keyword, the default is used. You can specify the SORTWORK keyword as follows:

```
SORTWORK(n,ccc,unitName,dataclas,mgmtclas,storclas)
```
The SORTWORK parameters are positional, and they must be separated by commas.

Table 255  SORTWORK parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>number of SORTWKnn data sets (0 through 32). The default value is 3. If you specify 0, no data sets are allocated.</td>
</tr>
<tr>
<td>$cccc$</td>
<td>number of primary cylinders to allocate for each data set (1 through 9999). The default value is 100.</td>
</tr>
<tr>
<td>$unitName$</td>
<td>unit name for sort work data sets. Specify any valid unit name. The default value is determined by what you specified in the environment setup member.</td>
</tr>
<tr>
<td>$dataclas$</td>
<td>SMS data class construct. Specify any valid data class name. The default value is determined by what you specified in the environment setup member.</td>
</tr>
<tr>
<td>$mgmtclas$</td>
<td>SMS management class construct. Specify any valid management class name. The default value is determined by what you specified in the environment setup member.</td>
</tr>
<tr>
<td>$storclas$</td>
<td>SMS storage class construct. Specify any valid management class name. The default value is determined by what you specified in the environment setup member.</td>
</tr>
</tbody>
</table>

The SORTWORK keyword applies to each sort.

For the secondary index sorts, the utility dynamically allocates the sort work data sets ddnames in the following format:

`SRT\$xWKnn`

The variable $x$ is the sort task that is running (1, 2, or 3) and $nn$ is the number of sort work data sets.

If you specify SORTWORK(0,$x$,x) and the amount of data to be sorted is too large for an incore sort, you must include SRT$xWKnn DD statements in the JCL.

If you specify PARALLEL(YES), the Index Build function chooses the optimum number of sorts (as many as three).

**NOTE**

The installation defaults for your sort product might affect how the SORTWORK keyword works. If the results you get from the SORTWORK keyword are not what you expected, check the sort product’s options. To override the sort product’s options, add the appropriate DD statements for your sort product to the BMC job step. For DFSORT, add a DFSPARM DD statement; for SyncSort, add a SORTPARM DD statement.
The SPILL keyword is valid with the RELOAD command. This keyword is available in PLUSIN. It is not recommended to be used as a global option. Table 256 describes the keyword.

Table 256 SPILL keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store segments temporarily in a DASD data set while evaluating the SEQERROR parm.</td>
<td>SPILL (Y)</td>
</tr>
<tr>
<td>Store segments in virtual storage while evaluating the SEQERROR parm.</td>
<td>SPILL (N)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> No</td>
<td></td>
</tr>
</tbody>
</table>

The SPILL keyword is intended for very rare occasions when a database has extremely large database records such as a single root and all of its dependents) and is running with SEQERROR(SAVE) or SEQERROR(ACCEPT).

**NOTE**
Extremely large database records that are over 128 megabytes SPILL(YES) negatively effect performance and are not recommended.

SPILL(YES) is mutually exclusive with SEGPOS. For more information about SEGPOS, see page 535.

The SSID keyword is valid with the OREORG command. This keyword is available in PLUSIN; it is not available as a global option. Table 257 describes the keyword.

Table 257 SSID keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use “imsid” as the IMS subsystem to be used for the BMP to connect to the IMS control region.</td>
<td>SSID(imsid)</td>
</tr>
<tr>
<td><strong>Internal default:</strong> Use the IMSID from the IMS generation (DFSVCO00 module).</td>
<td></td>
</tr>
</tbody>
</table>
**STATS**

The STATS keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 258 describes the keyword.

Table 258 Image copy statistics keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accumulate and print free space analysis information during Image Copy function processing.</td>
<td>STATS(Y)</td>
</tr>
<tr>
<td>Do not accumulate and print free space analysis information during Image Copy function processing.</td>
<td>STATS(N)</td>
</tr>
<tr>
<td>Internal default: STATS(Y)</td>
<td></td>
</tr>
</tbody>
</table>

For more information, see “Monitoring and reporting options” on page 334.

**SWAP**

The SWAP keyword is valid with the REORG (Online Reorg function and Reorg function) and COPY commands. You can specify this option in a Reorg or Online Copy function global options module or in PLUSIN. Table 259 describes the keyword.

---

**WARNING**

Regardless of the value of the SWAP keyword, the Online Reorg function and the Reorg function always require exclusive control during the data set name swapping process. If you attempt a multiple step JCL allocation, exclusive control is lost and the database becomes unavailable.

---

Table 259 Data set name swapping keyword (part 1 of 2)

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reorganize the database to a shadow database. Do not swap the data set names after a successful reorganization or online copy.</td>
<td>SWAP(NO)</td>
</tr>
<tr>
<td>Reorganize the database to a shadow database. Swap the data set names after a successful reorganization or online copy.</td>
<td>SWAP(YES)</td>
</tr>
<tr>
<td>Reorganize the database to a shadow database. Swap the data set names after a successful reorganization, and delete the old data sets. SWAP(DELETE) is forced to SWAP(YES) if you specify OLDBD and shorten the segment length.</td>
<td>SWAP(DELETE)</td>
</tr>
</tbody>
</table>
Chapter 14  User-controlled options  549

The SWAP keyword specifies how to activate the reorganized (shadow or image copy) data sets after a successful reorganization. For any value other than SWAP(NO), DBRC must be active and the data sets you are reorganizing must be registered. You can take any of the following actions:

- Delay the data set name swapping to a later job step. Specify SWAP(NO).
- Rename the shadow database to the original data set names (swap the data set names). Specify SWAP(YES) or SWAP(D).
- Restore a reorganized image copy of the database. Specify SWAP(YES,IC) or SWAP(S,IC).

The SWAP keyword is not valid when you specify INPUT(UNLDFILE).

You can set the default value for the SWAP option by changing the FRF$GLBL global options module. Sample JCL is in member #FRFGLBL of the sample library.

Whether you have the Reorg function swap the data set names or you swap them manually, the original (disorganized) data sets and the reorganized (shadow) data sets must be in the same MVS catalog. When the data sets are reorganized, they are renamed to have the original data set names. If the data sets are in different MVS catalogs, MVS would not be able to find the renamed data sets. The best way to ensure that the data sets are in the same catalog is to use the same high-level qualifier for both sets of data set names.

### Table 259  Data set name swapping keyword (part 2 of 2)

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>The following options are not available with MAXM Reorg/EP Express.</td>
<td></td>
</tr>
<tr>
<td>Reorganize the database to an image copy. After a successful reorganization, restore the image copy to the original data sets (Reorg function).</td>
<td>SWAP(YES,IC)</td>
</tr>
<tr>
<td>Reorganize the database to an image copy. After a successful reorganization, suspend the image copy restore (Reorg function).</td>
<td>SWAP(SUSPEND,IC)</td>
</tr>
<tr>
<td>Internal Default: NO</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE**

BMC recommends that you specify SWAP(YES).
Replicating a database

In most cases, the Reorg function will not execute if the original data sets and the shadow data sets are in different MVS catalogs. However, if you want to replicate the database, and leave the original database disorganized, you can have the original and reorganized data sets in different MVS catalogs. To replicate the database, follow these steps:

1. Specify SWAP(NO).

2. Specify AMSOUT DD DUMMY, or ensure that an AMSOUT file is not allocated for the job step. An AMSOUT file can be allocated by the FRF$GLBL global options module or a DD statement.

Restarting the reorganization

If you need to execute the restart process (you are swapping data set names manually or any post-processing task fails), an AMSOUT file is required.

**WARNING**

If you specify SWAP(N) or SWAP(S,IC), you must specify an AMSOUT data set. This file is required for the restart process. Do not modify the AMSOUT file.

Delay data set name swapping

Specify SWAP(NO) to delay data set name swapping. SWAP(NO) must be specified for the Reorg function if the reorganized database contains logical relationships and the prefix resolution and update function has not been integrated into the Reorg function execution.

**NOTE**

The DD cards DFSURWF1 and PRPURWF1 determine whether the Reorg function should integrate the execution of the prefix resolution and update function during the database reorganization.

If you specify SWAP(NO), you are responsible for ensuring that the swap takes place after the reorganization. You must (through a separate process) obtain exclusive control of the database from DBRC, swap the data set names, and notify DBRC of the reorganization. For more information, see Appendix B, “Post-processing.”
The Online Reorg function supports a new EXIT that enables you to delay the swap and continue logging until a defined period of time. The EXIT module is DLIOWNDW. No keyword is required to activate this feature. If the module is present, the feature is automatically activated. DLIOWNDW is documented in the sample library.

**Rename the shadow database**

During a Reorg function or a Online Reorg function reorganization, data sets are reorganized to shadow data sets. After a successful reorganization, the Reorg function and the Online Reorg function can rename the reorganized (shadow) data sets with the original (disorganized) data set names. This is called *data set name swapping*.

When you specify SWAP(YES), the primary database and primary index database data sets will be swapped. The secondary index databases will be swapped only if the data set specified in the JCL for the secondary index DBD ddname is different than what is registered to DBRC.

Automatic swapping requires that all of the following conditions be met:

- DBRC is active.
- All databases being reorganized, including the secondary indexes, are registered to DBRC.
- Primary database data sets used for output are not registered to DBRC.

---

**NOTE**

If the output secondary index data sets are registered to DBRC, they are excluded from swapping.

SWAP(D) is similar to SWAP(YES). The only difference is that when you specify SWAP(D), the original (disorganized) data sets are deleted.

---

**WARNING**

If you specify SWAP(D), you cannot save the disorganized database data sets. If you want a before-image backup, do not specify SWAP(D). You can also request a copy of the output unload file, but doing so degrades performance.
DFSURWF1 DD DUMMY is not allowed for automatic data set name swapping. If you specify DFSURWF1 DD DUMMY, secondary index and logical relationship records are ignored. When data set names are swapped automatically, the reorganized database is made available to users after the swap. You do not have a chance to resolve logical relationship prefixes before the database is made available; therefore, the reorganized data sets will be invalid. For more information, see the discussion of how the Reorg function treats the DFSURWF1 file on page 96.

Before swapping can occur, the Reorg function obtains exclusive authorization for the database from DBRC. If you specified SHARE(NO), authorization might have been obtained already. When the Reorg function has acquired exclusive authorization, it swaps the data set names and issues the NOTIFY.REORG command to DBRC. If you specify SWAP(D) or SWAP(YES), the Reorg function performs all these tasks automatically. BMC recommends that you allow the Reorg function to swap the data set names automatically.

If the Reorg function is allowed to do the swap, the original names of the disorganized data sets are temporarily appended with .T. The original data set name (including the VSAM component name) must not be longer than 42 characters regardless of whether you specify SWAP(N).

When the reorganized data sets have been renamed (to the DBRC-registered names), the Reorg function does one of two things:

- If you specified SWAP(YES), the Reorg function renames the .T version of the data set name to the name you specified for the reorganized data sets.
- If you specified SWAP(D), the Reorg function deletes the .T version of the data sets.

**Restore a reorganized image copy (Reorg function only)**

You can save DASD space by specifying SWAP(x,IC). When you specify SWAP(x,IC), the segments being reloaded are written directly to the image copy data sets and not to the shadow data sets. When all of the reorganized segments are written to the image copy, the image copy data sets are restored to the original database data sets. Because the secondary indexes are rebuilt directly to the original database data set names, the image copies of the secondary index data sets (if any) are not restored to the original secondary index data sets.
To use SWAP(x,IC), you must meet all of the following requirements:

- DBRC is required.
- You must specify ICP(YES).
- You must specify SHARE(NO) if the database has any secondary indexes.
- The database cannot have any logical relationships.
- You must include Image Copy function commands and keywords in an ICPSYSIN control statement data set to take an image copy of the primary database and the primary index (if HIDAM or PHIDAM).

To use SWAP(S,IC), you must ensure that one copy of each image copy data set is cataloged. The restart process requires cataloged image copy data sets.

The following options are not valid:

- SWAP(NO,IC)
- SWAP(D,IC). This value is forced to SWAP(YESE,IC).

If you specify two image copy data sets (one on DASD and one on tape), the copy that resides on DASD is the one that is restored.

**Restore image copy immediately**

When you specify SWAP(YES,IC), the database is reorganized to an image copy and not to shadow data sets. After the reorganization, the Reorg function restores the image copy to the original data set names. Use SWAP(YES,IC) if you want to stay on the same volume that the original database resides on. DFSURWF1 DD DUMMY is not allowed.

If you specify SWAP(YES,IC) and the primary and/or primary index is VSAM, you must specify REUSE. The secondary indexes can be deleted and defined before Reorg function execution and do not need to be defined as REUSE.

**NOTE**

The VSAM REUSE option is not honored for shared secondary indexes. Output shared secondary index data sets must be empty.

Secondary indexes are always rebuilt to the secondary index data set name as registered with DBRC (not to an image copy).
Suspend the image copy restore

When you specify SWAP(S,IC), the database is reorganized to an image copy and not to shadow data sets. The image copy restore is delayed for the primary database and primary index. Secondary indexes, if any, are not delayed because secondary indexes are built using the original data sets. They need to be deleted and defined before the reorganization. DFSURWF1 DD DUMMY is not allowed.

Use SWAP(S,IC) in the following situations:

- if you are changing the space allocations or reclaiming extents
- if your primary database VSAM data sets are not defined with REUSE

When you specify SWAP(S,IC), the image copy restore for the primary database and primary index is suspended until you execute the RESTART command or the Restart Swap utility. The Reorg function notifies DBRC to PROHIBIT AUTHORIZATION, and the primary databases are marked RECOV.NEEDED. After the restart process is executed successfully, the Reorg function notifies DBRC that recovery is no longer needed. However, the databases remain in NOAUTH status until you verify that the process is complete and you manually authorize the database by using DBRC.

If you specify SWAP(S,IC) and the primary databases are VSAM, you do not need to change your VSAM components to REUSE. The image copy restore is suspended for a later step, and you have a chance to delete and define the VSAM data sets before executing the restart process.

---

**NOTE**

Be careful when changing attributes of the original database data sets. The image copy must be restored to the original BLKSIZE, CISIZE, keys, and so on. (The most common data set change allowed would be to increase space.) BMC recommends that you reorganize to shadow data sets because of better performance.

Secondary indexes are always rebuilt to the secondary index data set name that is registered to DBRC.

---

SYNCTAPE

The SYNCTAPE keyword is valid with the UNLOAD command. You can specify this option in an Unload function global options module or in PLUSIN. Table 260 on page 555 describes the keyword.
Use the SYNCTAPE keyword to request synchronized tape volumes when you are making two copies of the unload data set on tape. When synchronized tape volumes are created, each volume of the first copy is interchangeable with the corresponding volume of the second copy.

**NOTE**
The Unload function does not support a mix of device types between the DFSURGU1 and DFSURGU2 DD statements.

Figure 91 shows the interchangeability of the volumes.

If the first tape volume from copy 1 and the third tape volume from copy 2 are unreadable, you still have a good unload file with these combinations:

- first and second tape volumes from copy 2 and third tape volume of copy 1
- first tape volume from copy 2 and the second and third tape volumes from copy 1

You can set the default value for the SYNCTAPE option by using the Unload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronize tape volumes when making two copies of the unload data set on tape.</td>
<td>SYNCTAPE (YES)</td>
</tr>
</tbody>
</table>

**Table 260  Synchronize tape volumes keyword**
**TIME**

The TIME keyword is optional on the GLBL command. This keyword is available in ICPSYSIN; it is also available in the global options module. Table 261 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provide a time stamp on Image Copy function messages in the processing log.</td>
<td>TIME(Y)</td>
</tr>
<tr>
<td>Do not provide a time stamp on Image Copy function messages in the processing log.</td>
<td>TIME(N)</td>
</tr>
</tbody>
</table>

**Internal default: TIME(N)**

The time stamps on Image Copy function messages in the processing log are available for performance testing and problem diagnosis. Use the TIME keyword to specify whether each message in the processing log will have a timestamp. For MVS/ESA, the task CPU time is appended to certain messages.

This option does not affect performance and can be helpful for performance testing and problem diagnosis. BMC recommends the use of TIME(Y) in all production and testing environments.

**TRUNC**

The TRUNC keyword is valid with the RELOAD, REORG (Online Reorg function and Reorg function), and GROUP commands. You can specify this option in a Reload function global options module or in PLUSIN. Table 262 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start each HIDAM or PHIDAM root in a new block.</td>
<td>TRUNC(YES)</td>
</tr>
<tr>
<td>Do not start each HIDAM or PHIDAM root in a new block.</td>
<td>TRUNC(NO)</td>
</tr>
</tbody>
</table>

**Internal default: NO**

You can specify that each HIDAM or PHIDAM root segment must start in a new block in the database. If you also specified that you do not want to search for space in a previous block with SCAN(nm), the root’s dependent segments follow in the same block and possibly into subsequent blocks. Figure 92 on page 557 shows an example.
Starting each root segment in a new block improves performance by significantly reducing the number of I/Os required when retrieving the root segment’s dependents. The database might require more disk space because the blocks are not being packed as tightly as they would be without the use of the TRUNC keyword. IMS might use this space after the reload. If you use the TRUNC keyword, reevaluate the use of the free space percentage factor keyword in the DBD.

### NOTE

Space left in the current block could be used by subsequent dependent segments using the SCAN keyword.

You can set the default value for the TRUNC option by using the Reload function ISPF panels or the equivalent batch job. Sample JCL is in member #DBUGLBL of the sample library.

### UCBITMAP

The UCBITMAP keyword is valid with the RELOAD and REORG commands. This keyword is available in PLUSIN; it is also available in the global options module. The UCBITMAP keyword may be used to control the setting of bitmap values that IMS uses after the reload to determine whether or not a block has free space. Table 263 on page 558 describes the keyword.
When this option is omitted, RELOAD/EP creates bitmaps indicating that all blocks contain space. You may use the UCBITMAP keyword to override this default.

You can specify a separate value for up to ten data set groups. The values in each sub parameter can be a null, an asterisk, or a one to five digit number. The values in the list correspond to the data set groups in the database. The first value is for the first data set group, etc. Table 264 lists the valid values for the UCBITMAP keyword and their meanings.

### Table 263  UCBITMAP keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set a user override value to use for the bitmap to indicate space available (per DSG).</td>
<td>(YES, value1, value2, ..., value10) or (value1, value2, ..., value10)</td>
</tr>
<tr>
<td>RELOAD/EP creates bitmaps indicating that all blocks contain space.</td>
<td>(NO)</td>
</tr>
</tbody>
</table>

Internal default: NO

When you reload the database with the IMS HD Reorganization Reload utility, the value used to build the bitmap is always equal to the largest segment in the data set group, even though most segments might be much smaller. When IMS tries to insert a segment into the database, the bitmap might show no free space in a block, even though the block might have enough space for the segment.

### Table 264  UCBITMAP values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number (1, ..., 32767)</td>
<td>Indicates that the bitmap for the corresponding data set group is to be reset using the specified number in determining whether there is space. The number must be in the range of 1 to 32767.</td>
</tr>
<tr>
<td>asterisk (*)</td>
<td>Indicates that the bitmap for the corresponding data set group is to be reset using the largest segment length (as specified in the DBD) in the data set group in determining whether there is space.</td>
</tr>
<tr>
<td>null (0)</td>
<td>Indicates that the corresponding data set group is not to be processed: i.e. the bitmaps for the data set group will be set to indicate that all blocks contain space.</td>
</tr>
</tbody>
</table>

A bit in the bitmap indicates whether a block has free space. If a block contains a contiguous area large enough to hold the largest segment in that data set group, it has free space. Table 265 lists the bitmap values.

### Table 265  Bitmap values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>free space in the bitmap</td>
</tr>
<tr>
<td>0</td>
<td>no free space in the bitmap</td>
</tr>
</tbody>
</table>
The value that you specify for controlling the Reload function bitmap is used only while building the bitmap during the reload process. After reloading, IMS uses its regular space management algorithm to manage bitmap blocks and to reset the bitmap bit for a block when IMS examines or uses the block.

For more information about bitmaps and how they work, refer to the BMC IMS Database Supplemental Utilities Reference Manual.

UNIT

The UNIT keyword is valid with the Index Build function and Prefix Resolution/Update function. You can specify this option in a global options module or in PLUSIN. Table 266 describes the keyword.

Table 266 Unit name keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the specified unit name for all dynamically allocated work data sets. (Index Build function)</td>
<td>UNIT(unitName)</td>
</tr>
<tr>
<td>Use the specified unit name when dynamically allocating the PRPURIDX work data set. (Prefix Resolution/Update function)</td>
<td>UNIT(unitName)</td>
</tr>
<tr>
<td><strong>Internal default:</strong></td>
<td></td>
</tr>
<tr>
<td>Index Build function: SYSDA</td>
<td></td>
</tr>
<tr>
<td>Prefix Resolution/Update function: TAPE</td>
<td></td>
</tr>
</tbody>
</table>

The UNIT option describes the unit name that the Index Build function uses for all dynamically allocated work data sets.

For the Prefix Resolution/Update function, specify the unit name to use when dynamically allocating the PRPURIDX work data set. If you specify SORTIDX(Y), you must also specify this option, even if you provide the PRPURIDX DD statement.

UNAUTH

The Unload Authorization (UNAUTH) keyword is valid with the UNLOAD command. You can specify this keyword in the environment setup member (DLIGSET0) or in PLUSIN. Table 267 on page 560 describes the keyword.
**UNLDD**

The UNLDD keyword is valid with the REORG (Reorg function) command. You can specify this option in a Reorg function global options module or in PLUSIN. Table 268 describes the keyword.

### Table 268 Input ddnames with substitution characters keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify input ddnames for primary databases.</td>
<td>UNLDD(ccc,n)</td>
</tr>
<tr>
<td></td>
<td>ccc represents a 1- to 3-character string.</td>
</tr>
<tr>
<td></td>
<td>n represents the position where you want the 1- to 3-character replacement string to appear.</td>
</tr>
<tr>
<td><strong>Internal default:</strong> OLD,1</td>
<td></td>
</tr>
</tbody>
</table>

The Reorg function requires a set of new ddnames to distinguish between input ddnames and output ddnames for primary database and primary index data sets. The Reorg function uses the ddnames defined in the DBD for the new data set names to be used for output. The Reorg function uses different ddnames for the old data sets that will be used for the unload process. This is the opposite of how a traditional (non-Reorg function) reorganization uses the DBD ddnames.

You must define the input ddnames in a way that is meaningful to you. The UNLDD keyword supplies edit information that defines the ddnames that the Reorg function uses for the original (disorganized) data sets.
The Reorg function uses the DBD ddnames as a base. The characters from the first value of UNLDD are placed into the ddname at the starting position specified by the second value. The characters overlay as much of the ddname as the length of the set of characters or the length of the ddname allows.

The ddnames created by the UNLDD edit must be unique and must not match those defined in DBRC for the database. If you did not specify DYNALLOC(Y,Y,x), the ddnames must appear in the job step JCL. The data sets assigned to these ddnames must be the original database data sets that are registered to DBRC because they are the data sets of the disorganized database to be used in the unload process.

You cannot use UNLDD and UNLDDLST in the same job step. These two keywords accomplish the same thing, but in different ways. UNLDD uses substitution characters; if you want to specify output and input ddnames as pairs (full ddname replacements) in a one-to-one list, use UNLDDLST. For more information, see “UNLDDLST” on page 562.

To ensure that you have the JCL set up correctly, the Reorg function checks the ddnames and the data set names paired with them.

**NOTE**

Do not specify DD statements for the old (original) secondary indexes when you use the UNLDD keyword; the Reorg function ignores these DD statements. For more information, see “Secondary indexes” on page 105.

The UNLDD keyword is not valid when you specify INPUT(UNLDFILE).

You can set the default value for the UNLDD option by changing the FRF$GLBL global options module. Sample JCL is in member #FRFGLBL of the sample library.

**Example:**

Database TESTDB is a HIDAM database containing two data set groups, TESTDD1 and TESTDD2. The primary index ddname is TESTDDX.

To define the ddnames TESTZZ1, TESTZZ2, and TESTZZX for the unload data sets, specify the following keyword with the REORG command:

```
UNLDD(ZZ,5)
```

Positions 5 and 6 of the ddname (DD in the original ddnames) are replaced with ZZ in each ddname.

Table 269 on page 562 shows examples for the UNLDD keyword.
The UNLDDLST keyword is valid with the REORG (Reorg function) and GROUP commands. This keyword is available in PLUSIN; it is not available as a global option. Table 270 describes the keyword.

Table 269  UNLDD example

<table>
<thead>
<tr>
<th>Old DBD ddname</th>
<th>Associated data set name</th>
<th>Substituted ddname</th>
<th>Associated data set name</th>
</tr>
</thead>
<tbody>
<tr>
<td>TESTDD1</td>
<td>new data set name for the primary database DSG1 that will be reloaded</td>
<td>TESTZZ1</td>
<td>the old (original) data set name for the primary database DSG1 that will be unloaded</td>
</tr>
<tr>
<td>TESTDD2</td>
<td>new data set name for the primary database DSG2 that will be reloaded</td>
<td>TESTZZ2</td>
<td>the old (original) data set name for the primary database DSG2 that will be unloaded</td>
</tr>
<tr>
<td>TESTDDX</td>
<td>new data set name for the primary index database that will be reloaded</td>
<td>TESTZZX</td>
<td>the old (original) data set name for the primary index database that will be unloaded</td>
</tr>
</tbody>
</table>

Table 270  Input ddnames in pairs keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify input ddnames for primary databases.</td>
<td>UNLDDLST((dbddnam1,xxddnam1), + (dbddnam2,xxddnam2))</td>
</tr>
<tr>
<td></td>
<td>dbddnam1 represents the ddname as defined in the DBD to be used for the new database to be loaded.</td>
</tr>
<tr>
<td></td>
<td>xxddnam1 represents a different ddname to be used for the original data set to be unloaded.</td>
</tr>
</tbody>
</table>

Internal default: none

The UNLDDLST keyword performs a function similar to that of the UNLDD keyword—it defines the ddnames for output and input into the Reorg function reorganization process.

The UNLDDLST keyword values make up a list of ddname pairs—each output (reload) ddname and its corresponding input (unload) ddname appear together. UNLDDLST is more flexible than UNLDD in the ddnames it can create, but you must specify all ddnames of a primary database (including the primary index and multiple data set groups) in the list.
The first value you specify with UNLDDLST keyword is the DBD ddname that will be used for the new output data set. The second value is the ddname that will be used for the associated unload data set. This is the opposite of how a traditional (non-Reorg function) reorganization uses the DBD ddnames.

---

**NOTE**

Do not specify secondary indexes with the UNLDDLST keyword; the Reorg function ignores secondary indexes specified with UNLDDLST. For more information, see “Secondary indexes” on page 105.

If you use the UNLDDLST keyword, the UNLDD keyword is not valid.

The UNLDDLST keyword is not valid when you specify INPUT(UNLDFILE).

**Example:**

Database TESTDB is a HIDAM database containing two data set groups, TESTDD1 and TESTDD2. The primary index ddname is TESTDDX.

The following example uses the UNLDDLST keyword to define the ddnames for the output (reorganized) data sets and the input (disorganized) data sets:

```plaintext
UNLDDLST((TESTDD1,MYDD1),(TESTDD2,MYDD2),+
 (TESTDDX,MYDDX))
```

All six ddnames must be included in the job step JCL for the Reorg function. You do not need to specify ddnames for the secondary indexes in UNLDDLST because the existing secondary index data sets are not used as input to the reorganization.

Table 271 shows examples for the UNLDDLST keyword.

---

### Table 271 UNLDDLST example

<table>
<thead>
<tr>
<th>Old DBD ddname</th>
<th>Associated data set name</th>
<th>New ddname</th>
<th>Associated data set name</th>
</tr>
</thead>
<tbody>
<tr>
<td>TESTDD1</td>
<td>new data set name for the primary database DSG1 that will be reloaded</td>
<td>MYDD1</td>
<td>the old (original) data set name for the primary database DSG1 that will be unloaded</td>
</tr>
<tr>
<td>TESTDD2</td>
<td>new data set name for the primary database DSG2 that will be reloaded</td>
<td>MYDD2</td>
<td>the old (original) data set name for the primary database DSG2 that will be unloaded</td>
</tr>
<tr>
<td>TESTDDX</td>
<td>new data set name for the primary index database that will be reloaded</td>
<td>MYDDX</td>
<td>the old (original) data set name for the primary index database that will be unloaded</td>
</tr>
</tbody>
</table>
UPDATE

The UPDATE keyword is valid with the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or in PLUSIN. Table 272 describes the keyword.

Table 272 Prefix update keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execute a concurrent prefix update during the output phase of DFSURG30/DFSURG10.</td>
<td>UPDATE(Y)</td>
</tr>
<tr>
<td>Do not execute a concurrent prefix update during the output phase of DFSURG30/DFSURG10.</td>
<td>UPDATE(N)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: Do not execute a concurrent prefix update.</td>
<td></td>
</tr>
</tbody>
</table>

Specify UPDATE(Y) to run a concurrent prefix update during the output phase of DFSURG30/DFSURG10. This completes the prefix update and prevents the creation of a DFSURWF3 data set for updated data sets. If you specify UPDATE(Y), concurrent prefix update is done for any data sets that have DD statements present; all other database data sets are written to DFSURWF3 or equivalent data sets. If you specify UPDATE(N), concurrent prefix update is not used and the work file data is written to DFSURWF3 or equivalent data sets.

USEINDEX

The USEINDEX keyword is valid with the DBSCAN and UNLOAD commands. This keyword is available in PLUSIN; it is not available as a global option. Table 273 describes the keyword.

Table 273 Unload roots using an index keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Force HIDAM database roots to be retrieved using the primary index.</td>
<td>USEINDEX(YES)</td>
</tr>
<tr>
<td>Retrieve HIDAM or HDAM database roots using a unique secondary index whose target is the root segment.</td>
<td>USEINDEX(dbdName)</td>
</tr>
<tr>
<td>Do not force root retrieval using an index.</td>
<td>USEINDEX(NO)</td>
</tr>
<tr>
<td><strong>Internal default</strong>: NO</td>
<td></td>
</tr>
</tbody>
</table>

Use the USEINDEX keyword to correct a broken/invalid PTF and PTB root pointer chain by forcing the roots to be accessed from an index.
Converting HDAM databases to HIDAM databases

The following keywords allow you to convert HDAM databases to HIDAM databases:

- Use the Index Build function to create a unique secondary index whose source is the root segment key and whose target is the root segment.
- Use the USEINDEX(dbdName) keyword to unload by using the named secondary index and to create a reload file in root key value sequence for a HIDAM reload.

Primary index

Specify USEINDEX(YES) to use the primary index to access the roots of a HIDAM database. Using this keyword is less efficient than following PTF and PTB chains (if they exist). Use the primary index only when the PTF and PTB chain is broken/invalid.

Secondary index

Secondary index root retrieval requires a unique index with direct pointers and a root segment target.

HDAM databases

For HDAM databases, the USEINDEX keyword will probably result in the unload and the reload doing random I/O for most of the roots, and both jobs will probably run longer for the average database. For very sparsely populated HDAM databases (less than 5 percent of RAPs used), the USEINDEX keyword eliminates scanning empty control intervals (CIs) and might improve the performance of the Unload function.

HIDAM databases

For HIDAM databases, you must sort the segments into key value sequence before reloading the database, if the source of the index is other than the root segment key field.

Consider the probable performance effects of unloading by using one of these indexes against the value of the result obtained.
USEINDEX limitations

The following limitations apply to the USEINDEX keyword:

- USEINDEX is supported only for HIDAM and HDAM.

- USEINDEX(dbdName) is not supported for HALDBs or the IMS/ESA Partition Support Product.

- Secondary indexes must meet the following criteria:
  - Secondary indexes must be unique; the index must consist of a single VSAM key-sequenced data set (KSDS).
  - Secondary indexes must have direct pointers rather than symbolic pointers.
  - The target must be the root segment.

- The Unload function does sequence checking by using the root key, not the secondary index key. If the key is different, many sequence error messages are issued.

**NOTE**

If the output of the unload is going to be input to a reload, you might want to add the restriction that the secondary index source is the root. This is not a BMC requirement. If the source is not contained in the root segment, you must ensure that one, and only one, index entry points to each root segment. If the database structure or the nature of the data ensures a one-to-one relationship of secondary index source to root segment, this is not a problem. Evaluate the performance impact of not unloading and reloading in RAP sequence for HDAM databases.

HIDAM unload performance is affected similarly. Reload performance is not affected because the load/reload process requires the data to be in sequence. This sequence is maintained if the primary index is used. Usage of any secondary index for a HIDAM reload and unload requires user intervention to preserve a one-to-one relationship and to sort the Unload function output into ascending sequence.

USEREXIT

The USEREXIT keyword is valid with the UNLOAD, RELOAD, REORG, and GROUP commands. You can specify this option in an Unload function or Reload function global options module or in PLUSIN. Table 274 on page 567 describes the keyword.
Chapter 14 User-controlled options

The Unload function, Reload function, Reorg function, and Online Reorg function support a user exit routine that allows you to examine, modify, and delete segments.

The Online Reorg function cannot use a user exit that was written for another product; it requires a special user exit.

Restrictions

The following restrictions apply to the use of a user exit:

**WARNING**

When moving the segment data from one area to another, make sure that you move only the length of the current segment.

- If you specify the USEREXIT keyword, databases are unloaded, reloaded, or reorganized in non-parallel mode.

- If you are removing a segment type from a DBD, you must delete all occurrences of the segment data from the HD Unload file before you reload the file with the Reload function or the Reorg function. You can use an Unload function user exit to delete the segment data.

### Table 274 User exit routine keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke a user exit routine.</td>
<td>USEREXIT(exitName)</td>
</tr>
<tr>
<td>Do not use a user exit routine.</td>
<td>Omit the keyword</td>
</tr>
<tr>
<td>Invoke an unload user exit routine during Reorg function or Online Reorg function processing.</td>
<td>USEREXIT(exitName,U)</td>
</tr>
<tr>
<td>Invoke a reload user exit routine during Reorg function or Online Reorg function processing.</td>
<td>USEREXIT(exitName,L)</td>
</tr>
<tr>
<td>Invoke a user exit routine while providing protection against abends due to a misunderstanding of the segment size.</td>
<td>USEREXIT(exitName,,YES)</td>
</tr>
<tr>
<td>Specifying YES copies the segment to a work area before calling the user exit routine.</td>
<td>USEREXIT(exitName,U</td>
</tr>
</tbody>
</table>

**Note:** If you specify NO or a blank for the third subparameter, the segment passed to the user exit remains in its original location. If the user exit tries to address data beyond the end of the segment, you might encounter a system S0C4 abend.

**Internal default:** none

TheUnload function, Reload function, Reorg function, and Online Reorg function support a user exit routine that allows you to examine, modify, and delete segments. The Online Reorg function cannot use a user exit that was written for another product; it requires a special user exit.
If you are using a user exit to change a key, execute that user exit when the database is unloaded. In the Reload function, if you use the HD Sort utility, the user exit is invoked after the HD Sort utility. Reload the database, and specify HDSORT(YES).

You can set the default value for the USEREXIT option by using the Unload function or Reload function ISPF panels or the equivalent batch job. Sample JCL is in the sample library.

USERHDR

The USERHDR keyword is valid with the RELOAD and UNLOAD commands. This keyword is available in PLUSIN; it is not available as a global option. Table 275 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create output records with a user-defined header.</td>
<td>USERHDR(subkeywords)</td>
</tr>
<tr>
<td>Use the standard header.</td>
<td>USERHDR()</td>
</tr>
<tr>
<td></td>
<td>or do not specify keyword</td>
</tr>
</tbody>
</table>

*Internal default*: None (use the standard header).

The USERHDR keyword tells the Unload function to create output records with a user-defined header (or prefix) in front of the segment data, and it tells the Reload function how to read the customized unload file.

**WARNING**

Use this option only if you have application programs that manipulate the DFSURGU1 file and change segment sequence before reloading the data. The USERHDR option does not improve performance. BMC recommends that you do not specify a USERHDR option unless you have programs that manipulate the DFSURGU1 file. Before you begin specifying USERHDR options, contact BMC Customer Support.

The USERHDR keyword is not valid with INPUT(UNLDFILE).

The USERHDR keyword is not valid when you unload or reload selected partitions of a database that uses the IMS/ESA Partition Support Product.
Unload function

With USERHDR, the Unload function writes an output data set (that the Reload function can read) with the following characteristics:

- The data set does not contain IMS header or trailer records such as those found in an HD Unload data set.
- The data set does not contain the Unload function header or trailer records (X'00'), which are used to verify keywords used between the unload and the reload.
- The data set contains one record per segment.
- Each record is variable length (for example, RECFM=VB); therefore, the QSAM record descriptor word (RDW) is included automatically.
- Each record consists of a header portion and a segment data portion.
- The format of the header portion is user defined.
- The segment data immediately follows the header. The segment data appears as seen by an application program, including the LPCK in front of the segment data.
- The Unload function writes the records to the data set defined by the DFSURGU1 DD statement. The Reload function reads the records from the data set defined by the DFSUINPT DD statement.

**NOTE**
You cannot use the Unload function USERHDR keyword and the DFSURGU1 keyword together.

- PHDAM and PHIDAM databases may be unloaded with the USERHDR keyword, but the unload files are not valid for RELOAD. They may be passed only to a user application.

Reload function

If you use the data set created under the USERHDR keyword in the Unload function as input to the Reload function to reload a database, the following rules apply:

- If your database has logical relationships, do not use USERHDR. The Unload function and the Reload function will run successfully, but prefix resolution might complete with errors.
If you use the USERHDR keyword for records that you unload with theUnloadfunction and reload with the Reload function, the values you code for theUSERHDR keyword must be identical for each utility.

If the data set does not contain the BMCFLAGS subkeyword and your databasehas segment compression defined, you must specify the correct compressionkeyword to match the status of the unloaded file as it was unloaded by the Unloadfunction when using the Reload function to load the database. BMC recommendsthat you specify the BMCFLAGS subkeyword to prevent the Reload function fromcompressing data already in compressed format.

Unload files of PHDAM or PHIDAM databases created with the USERHDRkeyword are not valid input to RELOAD.

Code the corresponding user header statement for the Reload function as shown inFigure 93.

Figure 93   User header statement for the Reload function

<table>
<thead>
<tr>
<th>LLZZ</th>
<th>user header</th>
<th>segment data</th>
</tr>
</thead>
<tbody>
<tr>
<td>↑ 4 characters</td>
<td>↑ variable length (contents are determined by the order and presence or absence of subkeywords)</td>
<td>↑ variable length</td>
</tr>
</tbody>
</table>

**WARNING**

You must specify the subkeywords in the same format and order as the corresponding fields in the user unload record. If you change the sequence or order of the header information between the unload and the reload steps, you must ensure that the subkeywords in the Reload function are correctly specified. If the subkeywords are incorrect, results are unpredictable.

**USERHDR subkeywords**

The USERHDR keyword has required and optional subkeywords that control which fields the Unload function writes and the Reload function reads in the user header.

**NOTE**

You must specify the SEGNAME or SEGCODE subkeyword; you can specify both. SEGCODE or SEGNAME must precede the SEGKEY or SEGCKEY fields.
You can use any combination of 18 subkeywords with the USERHDR keyword (maximum of 100 characters on one statement). Use the following syntax:

```
USERHDR(subkeyword, subkeyword, ...)
```

The order in which you specify the subkeywords determines the order in which the Unload function writes the fields in the user header. If you omit a subkeyword, the header field it controls will not appear in the output record.

To have the Unload function customize your unload file, or to have the Reload function read the customized unload file, code one or more of the subkeywords listed in Table 276.

### Table 276  USERHDR subkeyword descriptions and values

<table>
<thead>
<tr>
<th>USERHDR subkeyword</th>
<th>Length in bytes</th>
<th>Format of contents</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCFLAGS</td>
<td>1</td>
<td>binary</td>
<td>Reserved for BMC. Do not modify. Recommended field.</td>
</tr>
<tr>
<td>DBDNAME</td>
<td>8</td>
<td>character</td>
<td>Contains the DBD name.</td>
</tr>
<tr>
<td>FILLERnn</td>
<td>1-99</td>
<td>integer</td>
<td>The nn suffix defines length of the field. The field is filled with low-values (binary zeros).</td>
</tr>
<tr>
<td>HDRLEN</td>
<td>2</td>
<td>binary</td>
<td>Contains length of the user header record. This field is required if you use SEGKEY, SEGCKEY, or ROOTKEY.</td>
</tr>
<tr>
<td>ROOTKEY</td>
<td>variable varies</td>
<td>varies</td>
<td>Contains the root segment’s sequence field.</td>
</tr>
<tr>
<td>ROOTRBA</td>
<td>4</td>
<td>binary</td>
<td>Contains the root segment’s database RBA.</td>
</tr>
<tr>
<td>SEGCKYSZ</td>
<td>variable varies</td>
<td>binary</td>
<td>Contains segment’s fully concatenated key.</td>
</tr>
<tr>
<td>SEGCODE</td>
<td>2</td>
<td>binary</td>
<td>Contains the segment’s segment code. This keyword or SEGNAME is required.</td>
</tr>
<tr>
<td>SEGDATSZ</td>
<td>2</td>
<td>binary</td>
<td>Contains the length of the segment’s data.</td>
</tr>
<tr>
<td>SEGDBYTE</td>
<td>1</td>
<td>binary</td>
<td>Contains the segment’s delete byte as found in the database.</td>
</tr>
<tr>
<td>SEGKEY</td>
<td>variable varies</td>
<td>varies</td>
<td>Contains the segment’s sequence field.</td>
</tr>
<tr>
<td>SEGKEYOF</td>
<td>2</td>
<td>binary</td>
<td>Contains the offset into the segment data where the segment’s key is located.</td>
</tr>
<tr>
<td>SEGKEYSZ</td>
<td>2</td>
<td>binary</td>
<td>Contains the length of the segment’s key.</td>
</tr>
<tr>
<td>SEGLEVEL1</td>
<td>1</td>
<td>binary</td>
<td>Contains the segment’s hierarchical level.</td>
</tr>
<tr>
<td>SEGLEVEL2</td>
<td>2</td>
<td>character</td>
<td>Contains the segment’s hierarchical level.</td>
</tr>
<tr>
<td>SEGNAME</td>
<td>8</td>
<td>character</td>
<td>Contains the segment’s name. This keyword or SEGCODE is required.</td>
</tr>
<tr>
<td>SEGRBA</td>
<td>4</td>
<td>binary</td>
<td>Contains the segment’s database RBA.</td>
</tr>
</tbody>
</table>
Example control statement for the Unload function

Figure 94 shows the USERHDR keyword used with the UNLOAD command:

**Figure 94  USERHDR keyword example for the Unload function**

```
//PLUSIN DD *
UNLOAD USERHDR(DBDNAME, +
    SEGNAME, +
    SEGCODE, +
    SEGLEV1, +
    SEGRBA, +
    BMCFLAGS)
```

Example unload data set

If you use the examples shown Figure 94, the Unload function formats the header records as shown in Figure 95.

**Figure 95  Unload header record format**

```
<table>
<thead>
<tr>
<th>LLZZ</th>
<th>dbdname</th>
<th>segname</th>
<th>segcode</th>
<th>seglevel</th>
<th>segrba</th>
<th>bmcflags</th>
</tr>
</thead>
<tbody>
<tr>
<td>↑</td>
<td>↑</td>
<td>↑</td>
<td>↑</td>
<td>↑</td>
<td>↑</td>
<td>↑</td>
</tr>
</tbody>
</table>
```

(number of characters in record)

Example control statement for the Reload function

Figure 96 shows a control statement that can be used to read the record created with the example shown in Figure 94.

**Figure 96  USERHDR keyword example for the Reload function**

```
//PLUSIN DD *
RELOAD USERHDR(DBDNAME, +
    SEGNAME, +
    SEGCODE, +
    SEGLEV1, +
    SEGRBA, +
    BMCFLAGS)
```
The USERID keyword is optional on the GLBL command. This keyword is available in ICPSYSIN; it is also available in the global options module. Table 277 describes the keyword.

### Table 277  Image copy notify user ID keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify a list of one or more TSO users to notify when exceptional conditions occur during image copy processing.</td>
<td>USERID(userID1,userID2,...)</td>
</tr>
</tbody>
</table>

**Internal default:** USERID( )

For this keyword, an exceptional condition is defined as a non-zero return code or abend issued from the Image Copy function. (A non-zero return code or abend issued from POINTER CHECKER PLUS running under the Image Copy function would not be an exception condition.)

Each user ID value can be 1–8 alphanumeric characters. Specify a string of up to 99 characters. Separate the IDs with commas. Each comma counts as one character.

For more information, see “Monitoring and reporting options” on page 334.

The VIC keyword is optional on the GLBL, GROUP, IC, and AIC commands. This keyword is available in ICPSYSIN; it is also available in global and database-specific options modules. Table 278 describes the keyword.

### Table 278  Virtual image copy keyword

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in ICPSYSSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a virtual image copy of the primary and secondary indexes for the reorganized database.</td>
<td>VIC(Y)</td>
</tr>
<tr>
<td>Do not create a virtual image copy of the primary and secondary indexes for the reorganized database.</td>
<td>VIC(N)</td>
</tr>
</tbody>
</table>

**Internal default:** VIC(N)

A virtual image copy is not a real image copy; it is a dummy data set name recorded in DBRC as the image copy name of an index database. The dummy data set name must be cataloged.
You can create an image copy of only the first shared secondary index. Use the DBD keyword on the IC control statement. The Image Copy function does not support virtual image copies for shared secondary indexes.

The following restrictions apply when you specify VIC(Y):

- **SECONDARY INDEX UTILITY** must be installed.
- **DBRC** must be active (DBRC(Y)).
- The image copies defined to DBRC must be assigned the NOREUS parameter.

When you specify VIC(Y), you must also specify SIUDSN(*dataSetName*) in the global options module or on a control statement. For more information about this keyword, see “SIUDSN” on page 541.

For more information about virtual image copies, see “Virtual image copies” on page 330.

**XBMSSID**

The XBMSSID keyword is valid with the REORG (Online Reorg function) and UNLOAD commands. This keyword is available in PLUSIN; it is not available as a global option. Table 279 describes the keyword.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use “ssid” as the XBM subsystem to be used for the Snapshot Copy.</td>
<td>XBMSSID(<em>ssid</em>)</td>
</tr>
<tr>
<td>Use the XBMGROUP group name to use any XBM that is part of that XBMGROUP in the sysplex.</td>
<td>XBMSSID(<em>xbmGroupName</em>)</td>
</tr>
</tbody>
</table>

**Internal default**: None

If you have EXTENDED BUFFER MANAGER for IMS (XBM for IMS) or the SNAPSHOT UPGRADE FEATURE (SUF for IMS) installed and you specify SNAPSHOT(YES), XBM takes a Snapshot Copy of the database. The Snapshot Copy is used for the reorganization. This eliminates a full copy of the database, thus saving the DASD and the overhead associated with taking a copy. It also reduces the elapsed time of the reorganization.

When you specify SNAPSHOT(YES), you must also specify the name of the XBM subsystem to use for Snapshot Copy processing.
The XBM subsystem intercepts any updates made to an XBM-registered database, and it preserves the pre-image of the database. The Unload function includes the pre-image in the Snapshot Copy instead of the corresponding block read from the database.

**XFIXED**

The XFIXED option is available with the Prefix Resolution/Update function. You can specify this option in a Prefix Resolution/Update function global options module or in PLUSIN. Table 280 describes the option.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this in PLUSIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow fixed-length sort of index records.</td>
<td>XFIXED(Y)</td>
</tr>
<tr>
<td>Do not allow fixed-length sort of index records.</td>
<td>XFIXED(N)</td>
</tr>
<tr>
<td><strong>Internal default</strong> (N)</td>
<td></td>
</tr>
</tbody>
</table>

Specify XFIXED(Y) to allow fixed-length sorting of the index records. Fixed length sorts improve performance but might increase the sort work space required.

**MAXM Reorg keywords and the REORG command**

Some keywords that are valid with the BUILD, RELOAD, and UNLOAD commands are not valid with the REORG command. Other keywords that are valid with the BUILD, RELOAD, and UNLOAD commands can be coded on the REORG command. This section discusses these keywords.

**Keywords not available with the REORG command**

Some keywords that are available with the underlying functions and are not available with the REORG command. For example, you can code the COMPRESS keyword on the RELOAD command, but you cannot code it on the REORG command.

The Online Reorg function and the Reorg function assign the values they need to these unavailable keywords. If you need to use a different value, consider using the functions in stand-alone mode for databases in which reorganization performance would be affected by the values.
You cannot override the values the Online Reorg function and the Reorg function assign to these keywords.

Table 281 lists the keywords that are not available with the REORG command, which commands they are used with, and the value that the Online Reorg function and the Reorg function assign to them.

### Table 281  Keywords not available with REORG command

<table>
<thead>
<tr>
<th>Keyword</th>
<th>UNLOAD</th>
<th>RELOAD</th>
<th>BUILD</th>
<th>Reorg function value</th>
<th>Online Reorg function value</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPRESS</td>
<td></td>
<td></td>
<td>X</td>
<td>see page 415</td>
<td>(NO)</td>
</tr>
<tr>
<td>DBD</td>
<td></td>
<td>X</td>
<td></td>
<td>OS-Parm</td>
<td>OS-Parm</td>
</tr>
<tr>
<td>DBRC</td>
<td></td>
<td></td>
<td>X</td>
<td>OS-Parm</td>
<td>OS-Parm</td>
</tr>
<tr>
<td>DFSURGU1</td>
<td>X</td>
<td></td>
<td></td>
<td>(LONG)</td>
<td>n/a</td>
</tr>
<tr>
<td>GSGNAME</td>
<td></td>
<td></td>
<td>X</td>
<td>OS-Parm</td>
<td>OS-Parm</td>
</tr>
<tr>
<td>INDD</td>
<td></td>
<td></td>
<td>X</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>LIMITS</td>
<td>X</td>
<td></td>
<td></td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>NEWDBD</td>
<td>X</td>
<td></td>
<td></td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>PRIMIND</td>
<td></td>
<td></td>
<td>X</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>PTRCHECK</td>
<td>X</td>
<td></td>
<td></td>
<td>(NO)</td>
<td>n/a</td>
</tr>
<tr>
<td>PTRERROR</td>
<td>X</td>
<td></td>
<td></td>
<td>(ABEND)</td>
<td>n/a</td>
</tr>
<tr>
<td>SECINDEX</td>
<td></td>
<td></td>
<td>X</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>SEGSEL</td>
<td>X</td>
<td></td>
<td></td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>SEQERROR</td>
<td>X</td>
<td></td>
<td>X</td>
<td>(ABEND)</td>
<td>(ABEND)</td>
</tr>
<tr>
<td>SYNCTAPE</td>
<td>X</td>
<td></td>
<td></td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>USEINDEX</td>
<td>X</td>
<td></td>
<td></td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>USERHDR</td>
<td>X</td>
<td></td>
<td>X</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>

### Keywords available with the REORG command

Some keywords that are valid with the BUILD, RELOAD, and UNLOAD commands are also valid with the REORG command. The Online Reorg function and the Reorg function do not set values for these keywords.

If you do not supply the keyword with the REORG command, the usual process of determining keyword values (from defaults, global keywords, IMS-level keywords, DBD-level keywords, and so on) is used, with one exception: The only place that you can override the EXPAND keyword is on the REORG command.
An example of how to specify these keywords follows:

```
REORG SHARE(YES) SWAP(YES) IMSID(IMSP)
```

Table 282 lists the keywords that are valid on the REORG command.

### Table 282  Keywords from related utilities

<table>
<thead>
<tr>
<th>Keyword</th>
<th>UNLOAD</th>
<th>RELOAD</th>
<th>BUILD</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFSPACE</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CKUPDATE</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>EXPAND</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FBFF</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>FMTRAA</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSPF</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDAMFSPF</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HDSORT</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMSID</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>LDPPRINT</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>LPCK (Reorg function only)</td>
<td>see page 487</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OSAMMAX</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDX</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>PDXPARMS</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>REPORTS</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RMBYTES</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SCAN (space search)</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SEGPOS</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIUPRINT</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SORTWORK</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>TRUNC</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ULPPRINT</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEREXIT (Reorg function)</td>
<td>see page 671</td>
<td></td>
<td></td>
</tr>
<tr>
<td>USEREXIT (Online Reorg function)</td>
<td>see page 671</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Specifying default DBD libraries for Prefix Resolution/Update

This option is available with the Prefix Resolution/Update function. Table 283 describes the option.

Table 283  Default DBD libraries option

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this for ISPF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use this DBD library.</td>
<td>DBD Library =&gt; dbdName</td>
</tr>
<tr>
<td>Internal default: none</td>
<td></td>
</tr>
</tbody>
</table>

On the DBD Libraries panel specify the list of DBD libraries to be dynamically allocated if no IMS DD statement is present in the input JCL. The default DBD specifications can be used in all cases except for user-written initial load programs that use a PSB.

**WARNING**

If multiple IMS environments (multiple DBD libraries) can have the same DBD names, use extreme caution with this option.

If no DBD libraries are to be dynamically allocated, type **NONE** in the DBD Library field.
Reports

This chapter describes the reports generated by MAXM Reorg/Online and MAXM Reorg/EP solutions. This chapter includes the following topics:

Overview .................................................................................................................. 580
  Report destinations (most functions) ................................................................. 580
  Report destinations (Prefix Resolution/Update function) .......................... 581
  Report destinations (Image Copy function) .................................................... 581
  Reports stored in the PDX ................................................................. 581
Summary of reports ............................................................................................... 582
  ISPF interface to view reports ................................................................. 584
  Limit the number of report sets saved ...................................................... 584
Reports ...................................................................................................................... 585
  Bad Block/Pointer report ................................................................. 585
  Concurrent Reorganization ................................................................. 586
  Database Summary ............................................................................. 586
  Data Set Buffer Statistics report .......................................................... 587
  Data Set Group Summary ................................................................. 588
  Dataset List ......................................................................................... 591
  Dataset Statistics ............................................................................... 591
  Datasets Used ....................................................................................... 592
  DBD Change Summary ........................................................................ 592
  DBD Summary ....................................................................................... 592
  Disassembled DBD ............................................................................... 594
  Distribution of Database Record Sizes ................................................... 594
  Distribution of Free Space ..................................................................... 595
  Distribution of HDAM RAP Chain Lengths ........................................... 595
  Global Specifications ........................................................................... 596
  HDAM Root Addressable Area Statistics ............................................ 597
  HDAM Root Placement Relative to Its Home Block ................................ 598
  HD Sort - Record Counts report ......................................................... 599
  IDCAMS System Services ................................................................. 600
  Index Load Output Results .................................................................. 600
  Largest Database Records ..................................................................... 600
  Listing of PLUSIN Control Statements .................................................. 601
  Logical Relationship Information ....................................................... 601
  Logical Relationship Summary .......................................................... 601
Overview

When MAXM Reorg solutions terminate normally, they generate a set of reports. The statistics in these reports can help you make plans and decisions about future changes to your database.

Report destinations (most functions)

Most functions (except the Prefix Resolution/Update function and Image Copy function) send the reports to a certain destination as follows:

1. If you specified the xxxPRINT keyword, the reports are written to that data set. For more information, see "xxxPRINT" on page 522.

2. If you did not specify the xxxPRINT keyword but you specified the BMCPRINT DD statement, the reports are written to the BMCMSG data set.

3. If you did not specify the xxxPRINT keyword or the BMCPRINT DD statement, the functions that generate the report dynamically allocate the BMCPRINT data set.

You can generate Unload function reports without creating an unload data set by specifying a DFSURGU1 DD DUMMY statement in your JCL.
Report destinations (Prefix Resolution/Update function)

The Prefix Resolution/Update function sends the reports to the PLUSLIST data set if you included a PLUSLIST DD statement in the JCL. If you did not include a PLUSLIST DD statement, the Prefix Resolution/Update function dynamically allocates one to the default SYSOUT class. If the Prefix Resolution/Update function is unable to dynamically allocate a PLUSLIST data set, it writes reports to the ICPPRINT data set.

Report destinations (Image Copy function)

The Image Copy function generates reports and automatically sends them to the ICPPRINT data set unless you specify DUMMY on the ICPPRINT DD statement. If you include the REPORTS DD statement in the JCL, the Image Copy function writes the processing log to the ICPPRINT data set and the statistical reports to the REPORTS data set.

Reports stored in the PDX

To save the statistics in the PDX, you must allocate the PDX and specify a number greater than zero on the Report Parameters panel in the Number of sets per DBD field.

If you specified a default PDX name or a PDX DD statement in your JCL, the Unload function, the Reload function, the Index Build function, and the Image Copy function also store the reports in the PDX data set. You can access the reports stored in the PDX in the following ways:

- You can use the ISPF interface to view, print, or copy the reports.
- You can use a batch program to print the reports. For more information, see the BMC IMS Database Supplemental Utilities Reference Manual.
- You can extract the statistics in these reports for use with other programs. For more information, see the BMC IMS Database Supplemental Utilities Reference Manual.

**NOTE**

Reports pertaining to HALDBs or the IMS/ESA Partition Support Product with a partitioned primary index cannot be stored in the PDX. HALDB report information is stored in the SYSOUT data set.
Summary of reports

Table 284 lists (in alphabetical order) the reports generated by MAXM Reorg solutions, the functions that generate the report, and the page where the report is described.

**Table 284  Reports (part 1 of 2)**

<table>
<thead>
<tr>
<th>Report</th>
<th>Online Reorg</th>
<th>Reorg</th>
<th>Unload</th>
<th>Reload</th>
<th>Index Build</th>
<th>Prefix Res.</th>
<th>Image Copy</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concurrent Reorganization</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>586</td>
</tr>
<tr>
<td>Database Summary</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>586</td>
</tr>
<tr>
<td>Data Set Buffer Statistics</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>587</td>
</tr>
<tr>
<td>Data Set Group Summary</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>588</td>
</tr>
<tr>
<td>Dataset List</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>591</td>
</tr>
<tr>
<td>Dataset Statistics</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>591</td>
</tr>
<tr>
<td>Datasets Used</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>592</td>
</tr>
<tr>
<td>DBD Change Summary</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>592</td>
</tr>
<tr>
<td>DBD Summary</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>592</td>
</tr>
<tr>
<td>Disassembled DBD</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>594</td>
</tr>
<tr>
<td>Distribution of Database Record Sizes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>594</td>
</tr>
<tr>
<td>Distribution of Free Space</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>595</td>
</tr>
<tr>
<td>Distribution of HDAM RAP Chain Lengths</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>595</td>
</tr>
<tr>
<td>Global Specifications</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>596</td>
</tr>
<tr>
<td>HDAM Root Addressable Area Statistics</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>597</td>
</tr>
<tr>
<td>HDAM Root Placement Relative to Its Home Block</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>598</td>
</tr>
<tr>
<td>HD Sort - Records Count Report</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>599</td>
</tr>
<tr>
<td>IDCAMS System Services</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>600</td>
</tr>
<tr>
<td>Index Load Output Results</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>600</td>
</tr>
<tr>
<td>Largest Database Records</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>600</td>
</tr>
</tbody>
</table>
### Table 284  Reports (part 2 of 2)

<table>
<thead>
<tr>
<th>Report</th>
<th>Online Reorg</th>
<th>Reorg</th>
<th>Unload</th>
<th>Reload</th>
<th>Index Build</th>
<th>Prefix Res.</th>
<th>Image Copy</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Listing of PLUSIN Control Statements</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>601</td>
</tr>
<tr>
<td>Logical Relationship Information</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>601</td>
</tr>
<tr>
<td>Logical Relationship Summary</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>601</td>
</tr>
<tr>
<td>Message Log</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>604</td>
</tr>
<tr>
<td>Options in Effect</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>604</td>
</tr>
<tr>
<td>Output Data Set Summary</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>605</td>
</tr>
<tr>
<td>Output Image Copy Summary</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>605</td>
</tr>
<tr>
<td>Parent/Child Segment Statistics</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>606</td>
</tr>
<tr>
<td>Process Summary</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>607</td>
</tr>
<tr>
<td>Processing Log</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>607</td>
</tr>
<tr>
<td>Processing Summary</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>607</td>
</tr>
<tr>
<td>Reports for DBIL Type Reorganization</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>608</td>
</tr>
<tr>
<td>Reports for DBR Type Reorganization</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>608</td>
</tr>
<tr>
<td>Secondary Index Definition</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>609</td>
</tr>
<tr>
<td>Secondary Index Relationship Summary</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>611</td>
</tr>
<tr>
<td>Segment I/O Requirements Matrix</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>612</td>
</tr>
<tr>
<td>Segment Statistics</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>612</td>
</tr>
<tr>
<td>Status of Segments Read from HD Unload Data Set</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>613</td>
</tr>
<tr>
<td>Twin Segment Statistics</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>614</td>
</tr>
<tr>
<td>User Exit Routine Action</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>614</td>
</tr>
<tr>
<td>Variable Length Segment Statistics</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>615</td>
</tr>
<tr>
<td>VL Segment Split Statistics</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>616</td>
</tr>
</tbody>
</table>
ISPF interface to view reports

You can use the optional Database Utilities ISPF interface to view reports generated by the Unload function, the Reload function, the Index Build function, and the Image Copy function. The reports are stored in the PDX. MAXM Reorg solutions use the same ISPF interface as the classic database utilities. When a job step finishes processing, a set of reports is written to the PDX data set. The Online Reorg function and the Reorg function do not store reports in the PDX.

The PDX DD statement or the Default PDX data set option in the global options module determines which PDX contains the report sets. A user-specified option controls how many reports are saved for each IMSID/DBD/Report Type combination.

Each report set is distinguished by a member key that contains the IMSID of the IMS system, the DBD name of the database that was being processed, the type of job that generated the report, and the date and time the job was run.

Online help is available for each panel in the ISPF interface.

PDX is not supported for HALDBS. Report information for HALDBs is stored in the SYSOUT data set.

Limit the number of report sets saved

You can limit the number of reports saved in the PDX. Table 285 describes the options.

<table>
<thead>
<tr>
<th>To do this</th>
<th>Specify this for ISPF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit the number of reports sets saved in the PDX to nnn sets. (Unload function)</td>
<td>Specify the number of sets per DBD =&gt; nnn</td>
</tr>
<tr>
<td>Limit the number of reports sets saved in the PDX to nnn sets. (Reload function)</td>
<td>Number of sets per DBD =&gt; nnn</td>
</tr>
<tr>
<td>Limit the number of reports sets saved in the PDX to nnn sets. (Index Build function)</td>
<td>Statistics Set Limit =&gt; nnn</td>
</tr>
</tbody>
</table>

NOTE
The reports in Table 284 on page 582 that pertain to HDAM databases also pertain to PHDAM databases.
For each job, a set of statistical reports concerning the job is stored in the PDX. You can control the number of sets of reports each utility maintains per DBD by specifying a value for this option. You can keep a maximum of 999 reports for the Unload function, Reload function, and Index Build function. You can keep a maximum of 9999 reports for the Image Copy function. If you do not want to keep any reports, specify 0.

The reports limit value is examined each time a set of reports is stored in the PDX. If the number of report sets exceeds the current limit, the oldest report sets are deleted from the PDX.

### Reports

The reports are listed in alphabetical order. Each description tells which utility generated the report.

### Bad Block/Pointer report

If the Unload function encounters a bad pointer, it places a snap of the block containing the bad pointer in the ULPSNAP data set and a BMC90212 error message in the BMCMSG data set. (The ULPSNAP data set is dynamically allocated.) The code in the ERROR CODE field at the top of the report is explained in the description for message BMC90212. The SNAP ID value corresponds to the snap ID given in message BMC90212.

#### NOTE

For message explanations, access the BMC Documentation Center from the BMC Support Central site ([http://www.bmc.com/support](http://www.bmc.com/support)).

The Unload function prints the blocks containing the source RBAs and the blocks containing the target RBAs.
Concurrent Reorganization

The Concurrent Reorganization report summarizes the MAXM Reorg/Online run. The Online Reorg function generates this report automatically. The Concurrent Reorganization report provides the following information:

Table 286 Concurrent Reorganization report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGMENT NAME</td>
<td>The name of the segment in the database being reorganized.</td>
</tr>
<tr>
<td>UNLOAD COUNT</td>
<td>The segment count from the unload BMP. This field appears only if an unload BMP was used.</td>
</tr>
<tr>
<td>INSERTED</td>
<td>The number of inserts made to the online database while it was being reorganized.</td>
</tr>
<tr>
<td>DELETED</td>
<td>The number of DELETE calls made to the online database while it was being reorganized. This number does not reflect all segments deleted.</td>
</tr>
<tr>
<td>REPLACED</td>
<td>The number of replacements made to the online database while it was being reorganized.</td>
</tr>
<tr>
<td>SEGMENT CODE</td>
<td>The number assigned to a segment relative to its position in the database hierarchy.</td>
</tr>
<tr>
<td>CALL TYPE</td>
<td>The type of DL/I call issued.</td>
</tr>
<tr>
<td>STATUS CODE</td>
<td>The status code for each call type: BLANK (includes blank, GA, and GK status codes), GE, II, and OTHER (any other status code). <strong>Note</strong>: GNP calls are listed with GN calls. GHNP calls are listed with GHN calls.</td>
</tr>
<tr>
<td>TOTAL</td>
<td>The total number of calls per call type.</td>
</tr>
<tr>
<td>AVG. DUR (MSEC.)</td>
<td>The average duration of each DL/I call, in milliseconds.</td>
</tr>
<tr>
<td>TOTAL (1)</td>
<td>The total number of calls per segment type.</td>
</tr>
<tr>
<td>TOTAL (2)</td>
<td>The total number of calls for the database.</td>
</tr>
</tbody>
</table>

Database Summary

The Database Summary report lists the database-specific options in effect during the Image Copy function. The following fields are in the Database Summary report:

Table 287 Database Summary report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>The database name used in the Image Copy process.</td>
</tr>
<tr>
<td>ACCESS</td>
<td>The IMS access method defined for the database.</td>
</tr>
</tbody>
</table>
The Data Set Buffer Statistics report is generated by the Prefix Update function during prefix update processing. If you are running Prefix Update concurrently with the Logical Twin Resolution phase, this report appears under the phase title “Logical Twin Resolution.” The following fields are in the Data Set Buffer Statistics report:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSGS</td>
<td>The number of data set groups or areas defined for the database.</td>
</tr>
<tr>
<td>REGISTERED WITH DBRC</td>
<td>Indicates whether the database is registered with DBRC.</td>
</tr>
<tr>
<td>RANDOMIZER</td>
<td>The name of the HDAM or PHDAM randomizer model defined for the database.</td>
</tr>
<tr>
<td>RAPS</td>
<td>The number of root anchor points which are defined for each block of the database.</td>
</tr>
<tr>
<td>MAX RBN</td>
<td>The number of blocks defined in the DBD as the root addressable area for the database.</td>
</tr>
<tr>
<td>MAX BYTES</td>
<td>The maximum number of bytes as defined in the DBD that may be inserted into a root addressable block by a series of database ISRT calls.</td>
</tr>
<tr>
<td>keywords</td>
<td>The values that are in effect for this execution.</td>
</tr>
</tbody>
</table>

### Data Set Buffer Statistics report

The Data Set Buffer Statistics report is generated by the Prefix Update function during prefix update processing. If you are running Prefix Update concurrently with the Logical Twin Resolution phase, this report appears under the phase title “Logical Twin Resolution.” The following fields are in the Data Set Buffer Statistics report:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>The DBD name for the database being updated.</td>
</tr>
<tr>
<td>DDNAME</td>
<td>The ddname in the DD statement that describes the database data set.</td>
</tr>
<tr>
<td>DSORG</td>
<td>The data set organization of the database data set.</td>
</tr>
<tr>
<td>RECORDS</td>
<td>The number of logical relationship records that were updated.</td>
</tr>
<tr>
<td>READS</td>
<td>The number of blocks that were read from the database data set.</td>
</tr>
<tr>
<td>WRITES</td>
<td>The number of blocks that were written to the database data set.</td>
</tr>
<tr>
<td>UPDATES</td>
<td>The number of blocks that were updated.</td>
</tr>
</tbody>
</table>
Table 288  Data Set Buffer Statistics report fields (part 2 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%UPD</td>
<td>The percentage of read blocks that were updated (UPDATES divided by READS).</td>
</tr>
<tr>
<td>UPD CI</td>
<td>The average number of updates per control interval for VSAM, or the average number of updates per block for OSAM.</td>
</tr>
</tbody>
</table>

Data Set Group Summary

The Data Set Group Summary report shows information about each specific database data set group that was processed by the Image Copy function.

Table 289  Data Set Group Summary report fields (part 1 of 3)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDNAME</td>
<td>The ddname of the database data set group or the area name.</td>
</tr>
<tr>
<td>DSG</td>
<td>The data set group number.</td>
</tr>
<tr>
<td>DSNAMET</td>
<td>The data set name of this data set group or area.</td>
</tr>
<tr>
<td>BLKSIZE</td>
<td>The block size of this data set.</td>
</tr>
<tr>
<td>LRECL</td>
<td>The logical record length of this data set.</td>
</tr>
<tr>
<td>ORGANIZATION</td>
<td>The actual attributes of the data set (not what is defined in the DBD). These attributes can be OSAM or VSAM access methods and ESDS (the equivalent of a physical sequential OSAM data set) or KSDS organization. This field provides somewhat more specific information than is available from the DBD.</td>
</tr>
<tr>
<td>DEVICE</td>
<td>The device on which the data set group or area resides.</td>
</tr>
<tr>
<td>HIGH USED RBA</td>
<td>The highest relative byte address used by data in the data set.</td>
</tr>
<tr>
<td>HIGH ALLOCATED RBA</td>
<td>The highest relative byte address allocated to the data set.</td>
</tr>
<tr>
<td>BUFFERS USED</td>
<td>The number of buffers used when reading the data set group or area during the image copy. This field only applies when using native access methods.</td>
</tr>
<tr>
<td>VOLUME</td>
<td>The volume(s) on which the data set group or area resides. This field is only filled in when the Image Copy function actually accesses the database. For data sets that exceed seven volumes, information about only the first seven is stored in the PDX.</td>
</tr>
<tr>
<td>ACCESS METHOD USED</td>
<td>The access method that the Image Copy function actually used when processing the data set group or area.</td>
</tr>
<tr>
<td>SEGMENT SIZE RANGE FOR THIS DATA SET</td>
<td>The smallest and largest segment size (in bytes) for this data set group or area. Note that it is not the segment size range for the database.</td>
</tr>
<tr>
<td>BLOCKS IN ROOT ADDRESSABLE AREA</td>
<td>The number of blocks in the root addressable area in HDAM or PHDAM. The percentage is computed by dividing the number of blocks in the root addressable area by the number of blocks scanned. This value is generated only for HDAM and PHDAM primary data set groups.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NUMBER OF ROOT ANCHOR POINTS</td>
<td>The number of root anchor points which are presently available in the database. The number is computed by multiplying the number of root anchor points per block by the number of records in the root addressable area. This value is generated only for HDAM and PHDAM primary data set groups.</td>
</tr>
<tr>
<td>NUMBER OF UNUSED ROOT ANCHOR POINTS</td>
<td>The count of the root anchor points which have no data randomized to them. The percentage is computed by dividing the number of unused root anchor points by the total number of root anchor points. This value will be generated only for HDAM or PHDAM primary data set groups.</td>
</tr>
<tr>
<td>NUMBER OF BLOCKS INPUT</td>
<td>The count of the records that the Image Copy function read from the input database or dump data set.</td>
</tr>
<tr>
<td>NUMBER OF BLOCKS OUTPUT</td>
<td>The count of the records that were written to the output database. This line appears only when running the Database Recovery function. It is shown in the figure for illustration purposes only.</td>
</tr>
<tr>
<td>BYTES IN INPUT DATABASE</td>
<td>The total size of the database, including free space and unused blocks.</td>
</tr>
<tr>
<td>BYTES IN OUTPUT IMAGE COPY</td>
<td>The total size of the image copy data set, including the image copy prefix data. The number(s) do not include the size of the image copy header record. The percentage(s) at the right are computed by dividing the bytes value by the size of the output database, input database, or the input dump data set (whichever is appropriate). When image copying a KSDS or HISAM overflow database, the resultant image copy can be significantly larger than the input database. This occurs if the LRECL of the database is less than 64 bytes. The minimum LRECL for an output image copy data set is 64 bytes, which is the minimum size of the image copy header record. This restriction, which is similarly true for the IMS Database Image Copy utility, can cause the output image copy to be as much as 400 percent larger than the input database.</td>
</tr>
<tr>
<td>NUMBER OF BLOCKS SCANNED</td>
<td>The total number of blocks that were scanned during the space management analysis. This number might differ from the number of blocks input if blocks have been added to the end of the database during forward recovery or incremental image copy.</td>
</tr>
<tr>
<td>NUMBER OF UNUSED BLOCKS</td>
<td>The number of blocks that are completely unused (that is, contain no data or root anchor points). The percentage at the right is computed by dividing the number of unused blocks by the number of blocks scanned.</td>
</tr>
</tbody>
</table>
### Table 289  Data Set Group Summary report fields (part 3 of 3)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMBER OF BLOCKS CONTAINING FREE SPACE</td>
<td>The number of blocks which contain free space elements, not including blocks that are unused. The percentage is computed by dividing the number of blocks containing free space by the number of blocks scanned.</td>
</tr>
<tr>
<td>NUMBER OF COMPLETELY FREE BLOCKS</td>
<td>The number of blocks that previously contained data but contain no data now. It is one large free space element that begins immediately after the root anchor point and proceeds to the end of the record.</td>
</tr>
<tr>
<td>NUMBER OF FREE SPACE ELEMENTS</td>
<td>The count of the free space elements in the database, not including completely unused blocks.</td>
</tr>
<tr>
<td>TOTAL FREE SPACE IN FSES</td>
<td>The sum of the sizes of all of the free space elements (FSEs) in the database, not including any completely unused blocks. The percentage is computed by dividing the total space in FSEs by the size of the output database, input database, or input image copy (whichever is found first).</td>
</tr>
<tr>
<td>FSES THAT WILL HOLD LARGEST SEGMENT</td>
<td>The number of free space elements (FSEs) that are large enough to hold the largest segment size in the data set group or area. The percentage is computed by dividing the number of FSEs that will hold the largest segment by the total number of free space elements.</td>
</tr>
<tr>
<td>FSES TOO SMALL FOR SMALLEST SEGMENT</td>
<td>The number of free space elements (FSEs) that are too small to hold even the smallest segment in the database. The percentage is computed by dividing the number of FSEs that are too small by the number of records scanned.</td>
</tr>
<tr>
<td>SPACE IN FSES THAT ARE TOO SMALL</td>
<td>The total amount of space that is lost because it is in free space elements (FSEs) that are too small to hold any segment in the data set group or area. This space will remain inaccessible unless the segment that is immediately ahead of or behind it is deleted. The percentage is computed by dividing the space in FSEs that are too small by the size of the output database, input database, or input image copy (whichever is found first). Even small values in the percentage column can indicate that the database needs to be reorganized.</td>
</tr>
<tr>
<td>TOTAL FREE SPACE</td>
<td>The sum of the TOTAL FREE SPACE IN FSES and the SPACE IN UNUSED BLOCKS.</td>
</tr>
<tr>
<td>TOTAL USABLE FREE SPACE</td>
<td>The difference between the TOTAL FREE SPACE and the SPACE IN FSES THAT ARE TOO SMALL.</td>
</tr>
<tr>
<td>MINIMUM SIZE FREE ELEMENT</td>
<td>The size of the smallest free space element in the database.</td>
</tr>
<tr>
<td>MAXIMUM SIZE FREE ELEMENT</td>
<td>The size of the largest free space element in the database.</td>
</tr>
<tr>
<td>AVERAGE FSE SPACE PER BLOCK</td>
<td>The mean amount of free space in blocks that contain free space elements and the standard deviation from the mean.</td>
</tr>
<tr>
<td>AVERAGE PER CENT FREE SPACE PER BLOCK</td>
<td>The mean percentage of free space in blocks that contain free space elements and the standard deviation from the mean.</td>
</tr>
<tr>
<td>AVG SIZE OF FREE SPACE ELEMENT</td>
<td>The mean amount of space in free space elements and the standard deviation from the mean.</td>
</tr>
<tr>
<td>AVERAGE FREE SPACE ELEMENTS PER BLOCK</td>
<td>The mean percentage of space in a block that is in free space elements and the standard deviation from the mean.</td>
</tr>
</tbody>
</table>
NOTE
BMC provides the standard deviation to indicate the accuracy of the mean in representing the average size of the population. Small deviations indicate that the mean is relatively accurate, whereas larger deviations indicate that the individual sizes might vary considerably from the mean. For example, the mean of 50 and 50 is 50 and the standard deviation is zero, which shows that 50 accurately represents the sizes of the values in the population. The mean of 25 and 75 is also 50, but the standard deviation is 25, which indicates that the mean does not represent the actual population very well.

Dataset List

The Dataset List report provides the names of the input and output files. The Online Reorg function and the Reorg function generate this report automatically.

Dataset Statistics

The Dataset Statistics report provides the data set group (DSG) statistics. The Unload function and the Reload function generate this report automatically. The Dataset Statistics report provides the following information:

Table 290  Dataset Statistics report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSG</td>
<td>The data set group number for the row of information.</td>
</tr>
<tr>
<td>DDNAME</td>
<td>The ddname for the data set, obtained from the DBD.</td>
</tr>
<tr>
<td>BLOCK SIZE</td>
<td>The data set’s block/CI size.</td>
</tr>
<tr>
<td>RECORD SIZE</td>
<td>The data set’s logical record length.</td>
</tr>
<tr>
<td>DSNAME</td>
<td>The data set name used in the JCL.</td>
</tr>
<tr>
<td>FBFF</td>
<td>The free block frequency factor found in the DBD.</td>
</tr>
<tr>
<td>FSPF</td>
<td>The free space percentage factor found in the DBD.</td>
</tr>
<tr>
<td>FSPF BYTES</td>
<td>The number of bytes of space the free space percentage factor represents.</td>
</tr>
<tr>
<td>SMALL SEGSZE</td>
<td>The number of bytes of space required to hold the smallest segment in this data set group, including the prefix and data.</td>
</tr>
<tr>
<td>LARGE SEGSZE</td>
<td>The number of bytes of space required to hold the largest segment in this data set group, including the prefix and data.</td>
</tr>
<tr>
<td>TOTAL BLOCKS</td>
<td>The highest block/CI read in the data set.</td>
</tr>
</tbody>
</table>
Datasets Used

The Datasets Used report provides a list of the data sets used to build the indexes. The Index Build function generates this report automatically. The Datasets Used report provides the following information:

### Table 291  Datasets Used report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBDNAME</td>
<td>The name of the DBD that was used. The DBD can be a primary database, a primary index, or a secondary index.</td>
</tr>
<tr>
<td>DDNAME</td>
<td>The ddname that was used.</td>
</tr>
<tr>
<td>DSNAME FROM DBRC</td>
<td>The data set name registered to DBRC.</td>
</tr>
<tr>
<td>DSNAME IN JOB CONTROL</td>
<td>The data set name specified in the execution JCL.</td>
</tr>
</tbody>
</table>

DBD Change Summary

The DBD Change Summary report provides the results of a reorganization when you use the OLDDDBD option to change characteristics of the DBD during a reorganization.

The Reorg function generates this report when you specify the OLDDDBD option.

This report is identical to the DBD Summary report generated by the Unload function and the Reload function, except that this report shows values from the original and reorganized DBDs. The values from the original DBD are noted with OLD.

DBD Summary

The DBD Summary report shows the status and the specifications in the DBD at the time of the unload and the reload. The Unload function and the Reload function generate this report automatically. The DBD Summary report provides the following information:

### Table 292  DBD Summary report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE</td>
<td>The name of the database.</td>
</tr>
<tr>
<td>DS GROUPS</td>
<td>The number of data set groups.</td>
</tr>
<tr>
<td>PARTITIONS</td>
<td>The number of database partitions. This field is present only if the database uses the IMS/ESA Partition Support Product.</td>
</tr>
</tbody>
</table>
**Table 292  DBD Summary report fields (part 2 of 2)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORGANIZATION</td>
<td>The database organization.</td>
</tr>
<tr>
<td>ACCESS METHOD</td>
<td>The access method used.</td>
</tr>
<tr>
<td>RANDOMIZER</td>
<td>The name of the HDAM or PHDAM randomizer used.</td>
</tr>
<tr>
<td>NUMBER RAPS</td>
<td>The number of root anchor points (RAPs) in the HDAM or PHDAM parameters.</td>
</tr>
<tr>
<td>MAX RBN</td>
<td>The maximum RBNs in the RAA that was specified in the HDAM or PHDAM parameters.</td>
</tr>
<tr>
<td>MAX BYTES</td>
<td>The maximum number of bytes specified in the HDAM or PHDAM parameters that will be written to the RAA.</td>
</tr>
<tr>
<td>SEGMENT</td>
<td>The segment name.</td>
</tr>
<tr>
<td>SEG CODE</td>
<td>A one-byte hexadecimal value that identifies the segment in the database.</td>
</tr>
<tr>
<td>LVL</td>
<td>The segment’s hierarchical level.</td>
</tr>
<tr>
<td>DSG</td>
<td>The segment’s data set group.</td>
</tr>
<tr>
<td>PARENT</td>
<td>The name of the segment’s parent.</td>
</tr>
<tr>
<td>PFX LEN</td>
<td>The segment’s prefix length in bytes.</td>
</tr>
<tr>
<td>DATA LEN</td>
<td>The segment’s data length in bytes. If the segment is of variable length, this field displays the maximum data length.</td>
</tr>
<tr>
<td>FLAG</td>
<td>This field can contain one of the following values:</td>
</tr>
<tr>
<td></td>
<td>■ P—The segment was padded with one byte to make it an even length in the database. IMS does not write odd-length segments in HD-type databases. You could increase the segment length in the DBD by one byte and not use any more disk space.</td>
</tr>
<tr>
<td></td>
<td>■ V—The segment is defined as variable length.</td>
</tr>
<tr>
<td></td>
<td>■ C—The segment has an edit/compression routine.</td>
</tr>
<tr>
<td>PTR TYPE</td>
<td>The types of pointers contained in the segment’s prefix, listed in the same order as they appear in the segment’s prefix. If hierarchical pointers are used, the target segment type may vary depending on the other segments on the hierarchical chain.</td>
</tr>
<tr>
<td>TARGET SEGMENT</td>
<td>The name of the segment that the segment prefix pointer points to. For hierarchical pointers (HF and HB), the target segment might not always be known.</td>
</tr>
<tr>
<td>TARGET DBDNAME</td>
<td>The name of the database containing the target segment if this is a logical relationship pointer.</td>
</tr>
</tbody>
</table>
**Disassembled DBD**

When the DBD load module is disassembled, the resulting DBD source is printed in the Disassembled DBD report. Only the sequence field for each segment is displayed. The Unload function and the Reload function generate this report automatically.

In a partitioned HIDAM database, the HIGHKEY value is one byte shorter than the key length because IMS replaces the last byte with X'FF'.

**Distribution of Database Record Sizes**

The Distribution of Database Record Sizes report provides a frequency distribution of the database record sizes (prefix data plus segment data). The Unload function and the Reload function generate this report when you specify REPORTS(RCDDIST(YES)). To get a meaningful report, specify the distribution intervals with the RCDDIST option rather than use the default values. For more information, see “REORGMSC” on page 525.

The Distribution of Database Record Sizes report provides the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANGE</td>
<td>The user-specified minimum and maximum size in bytes of the interval. This size includes prefix and segment data.</td>
</tr>
<tr>
<td>FREQUENCY</td>
<td>The number of records with sizes that fit into the specified interval.</td>
</tr>
<tr>
<td></td>
<td>For databases with variable-length segments (including compressed segments defined as fixed length) where the data</td>
</tr>
<tr>
<td></td>
<td>length/compressed length is less than the minimum specified in the DBD, IMS writes the DBD minimum length</td>
</tr>
<tr>
<td></td>
<td>segment to DASD. The Reload function uses this actual segment length (DBD Minimum) to calculate the number of records</td>
</tr>
<tr>
<td></td>
<td>for each interval. The Unload function uses the data length for this calculation.</td>
</tr>
<tr>
<td>%</td>
<td>The number of records that fit into this interval, divided by the total number of records.</td>
</tr>
<tr>
<td>CUM%</td>
<td>The accumulated percentages. They are useful for statements about the percentage of records that are smaller than a</td>
</tr>
<tr>
<td></td>
<td>certain number of bytes. For example, 52.6 percent of the records are 201 to 300 bytes or less.</td>
</tr>
<tr>
<td>AVERAGE DATABASE</td>
<td>The average size (prefix plus segment data) of all database records, computed by using the actual segment size rather</td>
</tr>
<tr>
<td>RECORD SIZE</td>
<td>than the MIN specified in the DBD.</td>
</tr>
</tbody>
</table>
The Distribution of Free Space report provides the following information:

### Table 294  Distribution of Free Space report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSG</td>
<td>The data set group number for the row of information.</td>
</tr>
<tr>
<td>FREE SPACE RANGE</td>
<td>The lower and upper limits of the free space for this row in the report. The last row for each data set group always has a zero lower limit and an upper limit of one byte less than the smallest segment in the data set group. This last row shows the number of blocks that do not have enough free space to hold the smallest segment. The top row for each data set group always has a lower limit equal to the length of the largest segment in the data set group. The number of blocks shown on the top row should correspond to the number of blocks the bitmap shows as having free space. The rows between the top and bottom rows (if any) have lower and upper limits based on the sizes of the segments in that DSG. The numbers shown in these rows should help you determine how many blocks have enough free space to hold a particular segment type.</td>
</tr>
<tr>
<td>NUMBER BLOCKS</td>
<td>The number of blocks containing a free space element that falls within the free space range.</td>
</tr>
<tr>
<td>PCT OF TOTAL</td>
<td>The percentage of the total blocks that falls within the limits shown.</td>
</tr>
<tr>
<td>CUMULATIVE NBR</td>
<td>The total number of blocks that can hold segments of that size for any given free space interval.</td>
</tr>
<tr>
<td>BLOCKS</td>
<td></td>
</tr>
<tr>
<td>CUMULATIVE PCT TOTAL</td>
<td>The percentage of blocks that can hold a segment whose size fits in the free space interval.</td>
</tr>
</tbody>
</table>

### Distribution of HDAM RAP Chain Lengths

The Distribution of HDAM RAP Chain Lengths report shows the root synonym chain length and the number of chains. The Unload function generates this report automatically. The Reload function generates this report when you specify REPORTS(RAPCHAIN(YES)). For more information, see “REORGMMSG” on page 525.
When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition. The Distribution of HDAM RAP Chain Lengths report provides the following information:

**Table 295 Distribution of HDAM RAP Chain Lengths report fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAIN LENGTH</td>
<td>The number of roots that randomized to a particular RAP. A chain length of 1 indicates the number of roots that randomized to a unique RAP but were not actually on a RAP chain.</td>
</tr>
<tr>
<td>NUMBER OF CHAINS</td>
<td>The number of RAPs that have this chain length.</td>
</tr>
<tr>
<td>NUMBER OF ROOTS</td>
<td>The total number of roots involved in chains of this length.</td>
</tr>
<tr>
<td>AVERAGE ROOTS PER RAP</td>
<td>The total number of roots divided by the total number of RAPs.</td>
</tr>
<tr>
<td>MAXIMUM ROOTS PER RAP</td>
<td>The longest chain found.</td>
</tr>
</tbody>
</table>

**Global Specifications**

The Global Specifications report lists the global options in effect for the Image Copy function. It also reports selected environmental information.

**Table 296 Global Specifications report fields (part 1 of 2)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAPE DATE (ICP)</td>
<td>The date of the tape that contained the Image Copy function code.</td>
</tr>
<tr>
<td>CPU MODEL</td>
<td>The model of CPU running the Image Copy function at your installation.</td>
</tr>
<tr>
<td>CPU ID</td>
<td>The ID of your CPU.</td>
</tr>
<tr>
<td>OPERATING SYSTEM</td>
<td>The level, FMID, base release, and I/O Level (will be DFSMS or DFP Version) of the operating system used at your company.</td>
</tr>
<tr>
<td>IMSID</td>
<td>The value of the IMSID keyword.</td>
</tr>
<tr>
<td>IMS RELEASE LEVEL</td>
<td>The IMS release level of your system.</td>
</tr>
<tr>
<td>PARMBLK</td>
<td>The name of the internal global parameter block or the global options module.</td>
</tr>
<tr>
<td>PARAMETER BLOCK DESCRIPTION</td>
<td>The description assigned to the global parameter block or global options module.</td>
</tr>
<tr>
<td>keywords</td>
<td>The values of various Image Copy function keywords.</td>
</tr>
<tr>
<td>DBD LIBRARY NAME</td>
<td>The name(s) of the library(s) allocated to the IMS DD control statement.</td>
</tr>
</tbody>
</table>
The HDAM Root Addressable Area Statistics report shows information on the roots in an HDAM database. The Unload function and the Reload function generate this report automatically. When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition. The HDAM Root Addressable Area Statistics report provides the following information:

### Table 296  Global Specifications report fields (part 2 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STEPLIB CONCATENATION</td>
<td>The name(s) of the library(s) allocated to the STEPLIB DD control statement.</td>
</tr>
<tr>
<td>DBRC DATA SETS</td>
<td>The names of the RECON data sets used by DBRC.</td>
</tr>
</tbody>
</table>

### HDAM Root Addressable Area Statistics

The HDAM Root Addressable Area Statistics report shows information on the roots in an HDAM database. The Unload function and the Reload function generate this report automatically. When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition. The HDAM Root Addressable Area Statistics report provides the following information:

### Table 297  HDAM Root Addressable Area Statistics report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROOT SEGMENTS</td>
<td>The number of root segments (database records) in the database.</td>
</tr>
<tr>
<td>ROOT SYNONYM CHAINS</td>
<td>The number of HDAM RAPs with more than one root segment chained off of them.</td>
</tr>
<tr>
<td>ROOTS PER SYNONYM CHAIN</td>
<td>For RAPs with more than one root segment, the average number of root segments in the chain.</td>
</tr>
<tr>
<td>ROOTS NOT ON A SYNONYM CHAIN</td>
<td>The number of root segments that randomized to a unique RAP. No other roots randomized to this RAP.</td>
</tr>
<tr>
<td>ROOTS ON A SYNONYM CHAIN</td>
<td>The number of root segments that did not randomize to a unique RAP.</td>
</tr>
<tr>
<td>ROOTS IN LONGEST SYNONYM CHAIN</td>
<td>The largest number of root segments chained off a single RAP.</td>
</tr>
<tr>
<td>ROOT ANCHOR POINTS USED</td>
<td>The number of RAPs to which at least one root segment randomized.</td>
</tr>
<tr>
<td>ROOT ANCHOR POINTS NOT USED</td>
<td>The number of RAPs to which no root segments randomized.</td>
</tr>
<tr>
<td>ROOTS NOT IN THE RANDOMIZED BLOCK</td>
<td>The number of times a root segment randomized to a given block but the root was not found in the block. A relatively high number can affect performance.</td>
</tr>
</tbody>
</table>
The HDAM Root Placement Relative to Its Home Block report shows the distribution of HDAM roots relative to their home block (the home block is the block the root segment should be placed in, based on the randomizing routine).

The Unload function generates this report when you specify REPORTS(ROOTDIST(nn+mm)). The number of lines in this report are a function of the values specified in the ROOTDIST subkeyword. For more information, see “REORGMSG” on page 525.

### Table 297 HDAM Root Addressable Area Statistics report fields (part 2 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXIMUM CONSECUTIVE INSERT BYTES</td>
<td>The maximum number of bytes for a given HDAM database record placed in the RAA. When this value is reached during the load/reload process, subsequent segments for that database record are placed in the overflow area. This value is obtained from the DBD.</td>
</tr>
<tr>
<td>DB RECORDS EXCEEDING MAX INSERT BYTES</td>
<td>The number of HDAM database records with segment data that exceeded the maximum consecutive insert bytes value.</td>
</tr>
<tr>
<td></td>
<td>For databases with variable-length segments (including compressed segments defined as fixed length) where the data length/compressed length is less than the minimum specified in the DBD, IMS writes the DBD minimum length segment to the database.</td>
</tr>
<tr>
<td></td>
<td>The Unload function writes the data/compressed length segment to the output file (DFSURGU1) and uses this value to calculate the DB RECORDS EXCEEDING MAX INSERT BYTES count.</td>
</tr>
<tr>
<td></td>
<td>The Reload function output file is the database. The Reload function uses the DBD MIN when the data length is smaller, just like IMS. The Reload function also uses this value to calculate the DB RECORDS EXCEEDING MAX INSERT BYTES report item.</td>
</tr>
<tr>
<td></td>
<td>If the Reload function count of DB RECORDS EXCEEDING MAX INSERT BYTES is significantly larger than the count from the Unload function, this might indicate that DASD could be saved by decreasing the DBD MIN parameter for the variable length segments.</td>
</tr>
</tbody>
</table>
The Reload function generates this report when you use the Reload function HDAM space search algorithm and the database being loaded is an HDAM database. The report varies in size, based on the value you specified for the single limit value for the flip-flop scan or the forward and backward limits when you use the bidirected scan. For more information, see “SCAN—space search method for HDAM and PHDAM” on page 531.

When the database uses the IMS/ESA Partition Support Product, one report is generated for each partition. The HDAM Root Placement Statistics report provides the following information:

Table 298  HDAM Root Placement Statistics report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCATION</td>
<td>Location relative to the home block where HDAM roots should be placed based on the randomizing routine. Any roots found in the overflow area are also reported.</td>
</tr>
<tr>
<td>PCT</td>
<td>The percentage of roots placed in the associated relative block.</td>
</tr>
<tr>
<td>QUANTITY</td>
<td>The number of roots placed in the associated relative block.</td>
</tr>
</tbody>
</table>

**HD Sort - Record Counts report**

The HD Sort utility produces the HD Sort - Record Counts report to document the number of records it processed. The Reload function generates this report when you execute the HD Sort utility.

**NOTE**

If the HD Unload data set does not contain a valid trailer record, the HD Sort - Record Counts report is not generated.

The HD Sort - Record Counts report provides the following information:

Table 299  HD Sort - Record Counts report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGMENT CODE</td>
<td>The segment code from the HD Unload data set trailer record.</td>
</tr>
<tr>
<td>SEGMENT NAME</td>
<td>The name of the segment from the HD Unload data set trailer record.</td>
</tr>
<tr>
<td>COUNT FROM UNLOAD FILE</td>
<td>The number of segments of each type according to the HD Unload data set trailer record.</td>
</tr>
</tbody>
</table>
The IDCAMS System Services report lists the IDCAMS commands issued during data set name swapping. The Online Reorg function and the Reorg function generate this report automatically when you specify automatic data set name swapping.

### Index Load Output Results

The Index Load Output Results report lists the number of records written to the Index Build function work files for each index and the number of records suppressed. The Index Build function generates this report automatically. The Index Load Output Results report provides the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORDS WRITTEN TO INDEX</td>
<td>The total number of records written to the index.</td>
</tr>
<tr>
<td>EXCPS FOR INDEX</td>
<td>The number of EXCPs required for the index build process.</td>
</tr>
</tbody>
</table>

### Largest Database Records

The Largest Database Records report lists the \( nnn \) largest database records and the root’s RBA and key. A database record is a root segment plus its dependents. The records are listed in descending sequence, with the largest record first. The Unload function and the Reload function generate this report automatically. For more information, see “REORGMSG” on page 525.
When the database uses the IMS/ESA Partition Support Product, one report is generated for each partition. The Largest Database Records report provides the following information:

### Table 301 Largest Database Records report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORD SIZE (BYTES)</td>
<td>The size of the database record in bytes. A database record is a root segment and its dependents. This size includes all segment prefixes.</td>
</tr>
<tr>
<td>NBR SEGS IN RECORD</td>
<td>The number of segments in the database record.</td>
</tr>
<tr>
<td>ROOT’S RBA</td>
<td>The relative byte address of the root segment. For HISAM databases, this field is blank because HISAM databases do not use RBAs.</td>
</tr>
<tr>
<td>ROOT’S KEY</td>
<td>The first 22 bytes of the key of the root segment. If the root’s key contains any unprintable character (any character less than X’40’), the entire key is printed in hexadecimal (two lines per record).</td>
</tr>
</tbody>
</table>

### Listing of PLUSIN Control Statements

The Listing of PLUSIN Control Statements report shows which commands and keywords were included in the PLUSIN control statement data set. The Prefix Resolution/Update function generates this report.

### Logical Relationship Information

If your database has logical relationships, the Unload function and the Reload function generate the Logical Relationship Information report.

### Logical Relationship Summary

The Logical Relationship Summary report contains data about a logical relationship defined in the control data set (CDS) created by the IMS Prereorganization utility (DFSURPR0). A logical relationship is defined as a pair of segments where one segment is a logical child segment and the other is a logical parent segment. Logical parent segments can participate in multiple logical relationships and might appear in multiple Logical Relationship Summary reports. The report is generated by the Prefix Resolution and Update function and the Unload function.
The Logical Relationship Summary report is written for each logical relationship that exists in the CDS for which data was found by the scan. For jobs with multiple logical relationships, multiple Logical Relationship Summary reports are written.

There might appear to be discrepancies between input and output counts in databases with multiple logical relationships. Many of these are not errors. For instance, during database load, if a logical parent segment participates in multiple logical relationships, an output record is written for each logical relationship. Logical child first and last records are eliminated during the parent-child resolution phase of prefix resolution, and logical twin forward and backward records are eliminated during logical twin resolution processing. The Logical Relationship Summary report provides the following information:

Table 302 Logical Relationship Summary report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RELATIONSHIP DATA</td>
<td></td>
</tr>
<tr>
<td>DBD NAME</td>
<td>The DBD name of the logical parent and the logical child.</td>
</tr>
<tr>
<td>SEGMENT NAME</td>
<td>The segment name of the logical parent and the logical child.</td>
</tr>
<tr>
<td>DDNAME (DSGROUP)</td>
<td>The ddname of the data set that contains the respective database segments. The ddnames for the logical parent segment and the logical child segment are given.</td>
</tr>
<tr>
<td>DDNAME (OUTPUT)</td>
<td>The ddnames of the data sets into which the logical relationship records for the respective segments are written. The ddnames for the logical parent segment and the logical child segment are given.</td>
</tr>
<tr>
<td>RECORD COUNTS</td>
<td></td>
</tr>
<tr>
<td>LOGICAL PARENTS</td>
<td>The count of the logical parent records, which were read and written for this logical relationship. If the logical parents’ physical database is being loaded, reloaded, or scanned, and the logical parent segment participates in multiple logical relationships, the number of input records might be less than the number of output records for database load, reload, or scan processing. If the logical parent segment is not physically located in the database being loaded, reloaded, or scanned, the number of input and output logical parent records might be zero. If the number of input records is nonzero, the number of output records should be a multiple of the number of input records, with the multiple representing the number of logical relationships represented in the control data set created by the IMS Prereorganization utility.</td>
</tr>
</tbody>
</table>
Logical Relationship Summary

Table 302  Logical Relationship Summary report fields (part 2 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGICAL CHILD FIRST</td>
<td>The count of the logical child first records, which were read and written for this logical relationship. These records are written only if the physical database in which the logical child segment resides is being reorganized (DBR) rather than initially loaded (DBIL). If the logical twin segment has logical twin backward pointers, these records are not created. This record type is eliminated during parent-child resolution processing.</td>
</tr>
<tr>
<td>LOGICAL CHILD LAST</td>
<td>The count of the logical child last records, which were read and written for this logical relationship. These records are written only if the physical database in which the logical child segment resides is being reorganized (DBR) rather than initially loaded (DBIL). This record type is eliminated during parent-child resolution processing.</td>
</tr>
<tr>
<td>LOGICAL CHILDREN</td>
<td>The count of logical child logical relationship records, which were read and written for this logical relationship. These records are written when the physical database in which the logical child segment resides is being loaded, reloaded, or scanned. The number of input records and output records should always be equal.</td>
</tr>
<tr>
<td>LOGICAL TWIN FORWARD</td>
<td>The count of the logical twin forward pointer records, which were read and written for this logical relationship. These records are written only if the physical database in which the logical child segment resides is being reorganized (DBR) rather than initially loaded (DBIL) and logical twin forward pointers are specified. The count of these records should be less than the number of logical child records; the difference is the number of logical parent records created during the processing of the logical parent’s physical database. This record type is eliminated during the logical twin resolution phase.</td>
</tr>
<tr>
<td>LOGICAL TWIN BACKWARD</td>
<td>The count of the logical twin backward pointer records, which were read and written for this logical relationship. These records are written only if the physical database in which the logical child segment resides is being reorganized (DBR) rather than initially loaded (DBIL) and logical twin backward pointers are specified. The count of these records should be less than the number of logical child records; the difference is the number of logical parent records created during the processing of the logical parent’s physical database. This record type is eliminated during the logical twin resolution phase.</td>
</tr>
<tr>
<td>TOTALS FOR RELATIONSHIP</td>
<td>The total of all of the record types selected for this particular logical relationship. The count of input and output records might be different as discussed in the preceding fields.</td>
</tr>
</tbody>
</table>
Message Log

The Message Log lists the options specified on the PLUSIN control statement and the messages that MAXM Reorg solutions issued during the job execution. Each function (except the Prefix Resolution/Update function) generates this listing automatically and places the listing in the BMCMSG data set.

Options in Effect

The Options in Effect report lists the options in effect for the DBD. The report provides the name of the option, the value, and where the value came from. The values can come from internal defaults, a global options module, or the PLUSIN control statement. Each function generates this report automatically.

The fields listed on the report vary depending on which utility generates the report. The Options in Effect report provides the following information:

Table 303 Options in Effect report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS ID USED</td>
<td>The IMSID specified in your IMS stage 1 generation or with the IMSID keyword.</td>
</tr>
<tr>
<td>GLOBAL PARAMETER BLOCK USED</td>
<td>The member name of the global options module loaded from STEPLIB that provided the default option settings.</td>
</tr>
<tr>
<td>PDX DSNAME</td>
<td>The data set name of the PDX.</td>
</tr>
<tr>
<td>PDX MEMBER NAME</td>
<td>The name of the member within the PDX.</td>
</tr>
<tr>
<td>PDX HISTORY LIMIT</td>
<td>The number of report sets that the utility should save for each DBD.</td>
</tr>
<tr>
<td>REPORT DD</td>
<td>The ddname of the data set where reports are stored.</td>
</tr>
<tr>
<td>options</td>
<td>A list of the options in effect for this job. For more information, see Chapter 14, “User-controlled options.”</td>
</tr>
<tr>
<td>EXECUTION MODE OPTIONS</td>
<td>The remaining fields provide information about the job.</td>
</tr>
</tbody>
</table>
Output Data Set Summary

The Output Data Set Summary report is generated by the Unload function and the Prefix Resolution and Prefix Update function for each scan step. The report is a summary of all output data sets created as a result of the process. There is one report line for each output data set created. The Output Data Set Summary report provides the following information:

Table 304  Output Data Set Summary report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORDS</td>
<td>The count of the records written to the output data set.</td>
</tr>
<tr>
<td>DDNAME</td>
<td>The ddname in the DD statement that describes the output data set. The ddname can be used to relate the output data set to the logical relationship records and secondary index records written to the data set. The DDNAME field corresponds to the DDNAME (OUTPUT) field in the Logical Relationship Summary Report and the OUTPUT DDNAME field of the Secondary Index Relationship Summary Report.</td>
</tr>
<tr>
<td>DATA SET NAME</td>
<td>The fully qualified data set name for the data set that was created.</td>
</tr>
</tbody>
</table>

Output Image Copy Summary

The Image Copy function prints an Output Image Copy Summary report for each output image copy data set that is processed by the Image Copy function. The time stamp, when given, is normally the time stamp that appears on the data set.

Table 305  Output Image Copy Summary report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE</td>
<td>The image copy type (BMC BATCH).</td>
</tr>
<tr>
<td>ACCESS METHOD</td>
<td>The access method used to process the data set.</td>
</tr>
<tr>
<td>COPY #</td>
<td>The name(s) of the output image copy data set(s).</td>
</tr>
<tr>
<td>RECS</td>
<td>The total number of records in the data set.</td>
</tr>
<tr>
<td>BLOCKSIZE</td>
<td>The block size of the data set.</td>
</tr>
<tr>
<td>BUFFERS</td>
<td>The number of buffers used to process the data set.</td>
</tr>
<tr>
<td>DEVICE</td>
<td>The device type on which the data set resides.</td>
</tr>
<tr>
<td>ON VOLUME</td>
<td>The name of the volume(s) on which the data set is located.</td>
</tr>
</tbody>
</table>
Parent/Child Segment Statistics

The Parent/Child Segment Statistics report contains information about each parent/child relationship. The Unload function and the Reload function generate this report automatically. When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition. The Parent/Child Segment Statistics report provides the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARENT NAME</td>
<td>The parent segment’s name.</td>
</tr>
<tr>
<td>CHILD NAME</td>
<td>The child segment’s name.</td>
</tr>
<tr>
<td>NUMBER OF PARENTS WITH CHILD</td>
<td>The number of occurrences of this parent segment with this child segment.</td>
</tr>
<tr>
<td>NUMBER OF PARENTS W/O CHILD</td>
<td>The number of occurrences of this parent segment without this child segment.</td>
</tr>
</tbody>
</table>

Partition Stats

The Partition Stats report is generated if the database uses the IMS/ESA Partition Support Product. This report is generated instead of the Dataset Statistics report. The Partition Stats report provides the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRT</td>
<td>The partition number.</td>
</tr>
<tr>
<td>DDNAME</td>
<td>The ddname for the data set, obtained from the DBD.</td>
</tr>
<tr>
<td>BLOCK SIZE</td>
<td>The data set’s block/CI size.</td>
</tr>
<tr>
<td>RECORD SIZE</td>
<td>The data set’s logical record length.</td>
</tr>
<tr>
<td>DSNAME</td>
<td>The data set name used in the JCL.</td>
</tr>
<tr>
<td>FBFF</td>
<td>The free block frequency factor found in the DBD.</td>
</tr>
<tr>
<td>FSPF</td>
<td>The free space percentage factor (FSPF) found in the DBD.</td>
</tr>
<tr>
<td>FSPF BYTES</td>
<td>The number of bytes of space the FSPF represents.</td>
</tr>
<tr>
<td>SMALL SEGSZE</td>
<td>The number of bytes of space required to hold the smallest segment in this data set group, including prefix and data.</td>
</tr>
<tr>
<td>LARGE SEGSZE</td>
<td>The number of bytes of space required to hold the largest segment in this data set group, including prefix and data.</td>
</tr>
<tr>
<td>TOTAL BLOCKS</td>
<td>The highest block/CI written in the data set. If this is a VSAM data set, it includes CI 0. This number can help determine subsequent space allocation.</td>
</tr>
</tbody>
</table>
Process Summary

The Process Summary report is generated for each Prefix Resolution/Update function phase processed. The report is a summary of all input and output for each phase. If multiple phases are processed during the job step, multiple Process Summary reports are generated.

Because the Prefix Resolution/Update function does not capture data for secondary indexes during load or reload processing, no secondary index data is reported for DFSDSEH0. The following fields are in the Process Summary Information report:

Table 308 Process Summary Information report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGICAL RELATIONSHIPS</td>
<td>A total of the logical relationship records that were processed during this Prefix Resolution/Update function phase. If no logical relationships were processed during this phase, the counts will be zero.</td>
</tr>
<tr>
<td>SECONDARY INDICES</td>
<td>A total of the secondary index records that were processed during this Prefix Resolution/Update function phase. If no secondary index records were processed during this phase, the counts will be zero. During load, reload, or scan, the secondary index record counts will be zero even if secondary index records were created. the Prefix Resolution/Update function does not capture data for secondary indexes except during the Work File Splitter program, the Secondary Index Resolution program, and the Parent-Child Resolution phase.</td>
</tr>
</tbody>
</table>

Processing Log

As the Image Copy function runs, it writes messages to the Processing Log report about actions taken during the run.

This report shows a listing of the input control statements from the ICPSYSIN control statement data set.

Any diagnostics that are relative to the control statements follow. DFS, DSP, and critical BMC messages are marked with an asterisk.
Processing Summary

The Image Copy function writes the Processing Summary report to summarize information about the database and data set group that were processed.

Table 309  Processing Summary report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBD</td>
<td>The database name of the database that was image copied.</td>
</tr>
<tr>
<td>DDN/AREA</td>
<td>The ddname of the database data set.</td>
</tr>
<tr>
<td>DSG</td>
<td>The number of data set groups or areas defined for the database.</td>
</tr>
<tr>
<td>CMD</td>
<td>In the sequence of commands specified in the ICPSYSIN data set, the number of the command that processed this database data set.</td>
</tr>
<tr>
<td>ORG</td>
<td>The IMS organization of the data set.</td>
</tr>
<tr>
<td>ACCESS TYPE</td>
<td>The IMS access method defined for the database.</td>
</tr>
<tr>
<td>ACTION</td>
<td>The image copy command (IC or AIC) that was performed for the database data set.</td>
</tr>
<tr>
<td>STATUS</td>
<td>The results of processing.</td>
</tr>
</tbody>
</table>

Reports for DBIL Type Reorganization

The DBIL Type Reorganization report has the following fields. The report is generated by the Prefix Resolution and Prefix Update function.

- The length of the record in hex.
- The ZZ field reserved by IBM.
- The RBA of the logical parent segment in the database.
- The internal flags in the Prefix Resolution/Update function.
- The RBA of the logical child segment in the database.
- The LPCK.
- The filler to print an entire fullword.
Reports for DBR Type Reorganization

The DBR Type Reorganization report has the following fields. Each is generated by the Prefix Resolution and Prefix Update function.

- The length of the record in hex.
- The ZZ field reserved by IBM.
- The RBA of the logical parent segment in the database.
- The internal flags in the Prefix Resolution/Update function.
- The LP counter field in the logical parent segment.
- The internal flags in the Prefix Resolution/Update function.
- The old RBA of the logical parent segment before reorganization.
- The filler to print an entire fullword.
- The RBA of the logical child segment in the database.
- The LP counter field in the logical parent segment.

Secondary Index Definition

The Secondary Index Definition report lists the values stored in the DBD for the secondary indexes that were built. The Index Build function generates this report automatically. The Secondary Index Definition report provides the following information:

Table 310 Secondary Index Definition report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECONDARY INDEX DBD NAME</td>
<td>The DBD name of the secondary index.</td>
</tr>
<tr>
<td>TARGET DBD NAME</td>
<td>The name of the target DBD.</td>
</tr>
<tr>
<td>XDFLD NAME</td>
<td>The index field name specified in the source segment. It is the name used in the NAME operand of the XDFLD macro in the indexed DBD.</td>
</tr>
<tr>
<td>TARGET SEGMENT</td>
<td>The database segment pointed to by the secondary index pointer segment.</td>
</tr>
<tr>
<td>SOURCE SEGMENT</td>
<td>The database segment containing the data that the Index Build function uses to build the secondary index segment.</td>
</tr>
<tr>
<td>SHARED INDEX CONSTANT</td>
<td>A one-byte constant that identifies all pointer segments for a specific index in a shared index database. The value in the constant field is part of the index key. This field appears only for shared secondary indexes.</td>
</tr>
<tr>
<td>DATABASE DDNAME(S)</td>
<td>The ddname(s) of the database for which the secondary index was built.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SEARCH FIELD(S)</td>
<td>The following fields specify the data in the search field, which is the key to the secondary index pointer segment. All data in the search field comes from the source segment.</td>
</tr>
<tr>
<td></td>
<td>- NAME—The name used to identify the search field.</td>
</tr>
<tr>
<td></td>
<td>- START—The starting location of the search field.</td>
</tr>
<tr>
<td></td>
<td>- LENGTH—The length (in bytes) of the data in the search field.</td>
</tr>
<tr>
<td></td>
<td>- TYPE—The type of data in the search field.</td>
</tr>
<tr>
<td>SUBSEQUENCE FIELD(S)</td>
<td>The following fields specify the data in the subsequence fields, which the Index Build function uses to make non-unique keys unique:</td>
</tr>
<tr>
<td></td>
<td>- NAME—The name used to identify the subsequence field.</td>
</tr>
<tr>
<td></td>
<td>- START—The starting location of the subsequence field.</td>
</tr>
<tr>
<td></td>
<td>- LENGTH—The length (in bytes) of the data in the subsequence field.</td>
</tr>
<tr>
<td></td>
<td>- TYPE—The type of data in the subsequence field.</td>
</tr>
<tr>
<td>DUPLICATE DATA FIELD(S)</td>
<td>The following fields specify the data in the duplicate data fields:</td>
</tr>
<tr>
<td></td>
<td>- NAME—The name used to identify the duplicate data field.</td>
</tr>
<tr>
<td></td>
<td>- START—The starting location of the duplicate data field.</td>
</tr>
<tr>
<td></td>
<td>- LENGTH—The length (in bytes) of the data in the duplicate data field.</td>
</tr>
<tr>
<td></td>
<td>- TYPE—The type of data in the duplicate data field.</td>
</tr>
<tr>
<td>NULLVAL</td>
<td>A one-byte hexadecimal value used to suppress entries in the secondary index database.</td>
</tr>
<tr>
<td>TYPE OF POINTER</td>
<td>The type of pointer used for the secondary index. Direct pointing points to a segment by using its physical address. Symbolic pointing points to a segment by using its key.</td>
</tr>
<tr>
<td>SET IMAGE COPY NEEDED FLAG</td>
<td>Whether the DBRC RECON data sets have the ICNEEDED flag turned on or off for this index DBD.</td>
</tr>
<tr>
<td>LAYOUT OF THE INDEX RECORD</td>
<td>The fields described in Table 311 on page 611 specify the location and the number of bytes used for each field in the secondary index pointer segment.</td>
</tr>
</tbody>
</table>
The Secondary Index Relationship Summary report is generated for each Secondary Index Resolution phase processed. It summarizes all secondary indexes selected by the scan, with one line of report data for each secondary index found. This report is generated by the Unload function and the Prefix Resolution and Prefix Update function. The Secondary Index Relationship Summary report provides the following information:

### Table 311  Index record fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n-BYTE DELETE BYTE</td>
<td>The delete byte and its location in the secondary index pointer segment. The delete byte determines when a segment was flagged for deletion from the database.</td>
</tr>
<tr>
<td>n-BYTE pointerType POINTER</td>
<td>The pointer fields and their location in the secondary index pointer segment. pointerType is DIRECT or SYMBOLIC.</td>
</tr>
<tr>
<td>n-BYTE SHARED INDEX CONSTANT</td>
<td>The shared index constant field and its location in the secondary index pointer segment.</td>
</tr>
<tr>
<td>n-BYTE SEARCH FIELD(S)</td>
<td>The search fields and their locations in the secondary index pointer segment.</td>
</tr>
<tr>
<td>n-BYTE SUBSEQUENCE FIELD(S)</td>
<td>The subsequence fields and their locations in the secondary index pointer segment.</td>
</tr>
<tr>
<td>n-BYTE DUPLICATE DATA</td>
<td>The duplicate data fields and their locations in the secondary index pointer segment.</td>
</tr>
<tr>
<td>n-BYTE FILLER</td>
<td>A null field used to make a segment’s length an even number of bytes.</td>
</tr>
</tbody>
</table>

### Secondary Index Relationship Summary

The Secondary Index Relationship Summary report is generated for each Secondary Index Resolution phase processed. It summarizes all secondary indexes selected by the scan, with one line of report data for each secondary index found. This report is generated by the Unload function and the Prefix Resolution and Prefix Update function. The Secondary Index Relationship Summary report provides the following information:

### Table 312  Secondary Index Relationship Summary report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOURCE DBD</td>
<td>The physical database from which the data to construct the secondary index is obtained.</td>
</tr>
<tr>
<td>SOURCE SEGMENT</td>
<td>The physical database segment from which the data to construct the secondary index is obtained.</td>
</tr>
<tr>
<td>INDEXED FIELD</td>
<td>The XDFLD name specified in the source segment in the source database that describes the contents of the secondary index record.</td>
</tr>
<tr>
<td>TARGET SEGMENT</td>
<td>The physical database segment within the source database that is the target of the secondary index. This segment can be the source segment or any segment above the source segment in the source database’s physical hierarchy.</td>
</tr>
<tr>
<td>INDEX DBD</td>
<td>The physical database in which the secondary index data is stored.</td>
</tr>
</tbody>
</table>
Segment I/O Requirements Matrix

The Segment I/O Requirements Matrix report shows the number of I/Os required by DL/I to go from one segment type to another segment type. The rows and columns contain segment names. The dimension of the matrix is determined by the number of physical segment types in the database. The intersection of any row and column shows the number of I/Os required to go from the segment named in the row to the segment named in the column. Only segments that lie in a hierarchical path have a value in the matrix.

The probability is calculated by examining all nonzero pointers and determining whether they point within the same block as the source segment or into a different block. If the target and source segments are in different data set groups, the I/O probability is set to 1.000. You can transform the given I/O probability to a percentage by moving the decimal points two places to the right.

The Reload function generates this report automatically. The Unload function generates this report when you specify REPORTS(IOSTATS(YES)). For more information, see “REORGMSG” on page 525.

When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition.

Segment Statistics

The Segment Statistics report shows information about each segment type. The Unload function and the Reload function generate this report automatically. When the database uses the IMS/ESA Partition Support Product, one report is generated for each partition.

Table 312  Secondary Index Relationship Summary report fields (part 2 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT DDNAME</td>
<td>The ddname that represents the data set in which all output records were written.</td>
</tr>
<tr>
<td>INPUT RECORDS</td>
<td>The count of the records selected for this secondary index.</td>
</tr>
<tr>
<td>OUTPUT RECORDS</td>
<td>The count of the output records written for this secondary index. The count of output records should equal the input records.</td>
</tr>
</tbody>
</table>
The Segment Statistics report provides the following information:

**Table 313  Segment Statistics report fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGMENT</td>
<td>The segment’s name.</td>
</tr>
<tr>
<td>SEGMENT LENGTH</td>
<td>The segment’s prefix length plus the data length.</td>
</tr>
<tr>
<td>TOTAL NUMBER</td>
<td>The number of segment occurrences in the database.</td>
</tr>
<tr>
<td>NBR IN OVERFLOW</td>
<td>If it is an HDAM database, the number of this segment type found in the HDAM overflow area.</td>
</tr>
<tr>
<td>AVG OCCURRENCE PER PARENT</td>
<td>The average number of occurrences of this segment for each occurrence of its parent segment. This is the average twin chain length.</td>
</tr>
<tr>
<td>AVG OCCURRENCE PER RECORD</td>
<td>The average number of occurrences of this segment type per database record.</td>
</tr>
<tr>
<td>CUM SIZE</td>
<td>The cumulative average database record size as you move hierarchically through the database structure. The last number in the CUM SIZE column gives the length of the average database record.</td>
</tr>
<tr>
<td>TOTAL SEGMENTS</td>
<td>The total number of segments in the database and, if appropriate, in the overflow area. The segments in the overflow area are included in the count of total segments in the database.</td>
</tr>
</tbody>
</table>

**Status of Segments Read from HD Unload Dataset**

The trailer record for the HD Unload data set contains counts of each segment unloaded from the database. The Reload function compares these counts with its counts for each segment inserted into the database plus the number of segments deleted by a user exit. The Reload function generates the Status of Segments Read from HD Unload Dataset report automatically. The Status of Segments Read from HD Unload Dataset report provides the following information:

**Table 314  Status of Segments Read from HD Unload Dataset report fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGNAME</td>
<td>The segment’s name.</td>
</tr>
<tr>
<td>UNLOADED</td>
<td>The number of segments unloaded.</td>
</tr>
<tr>
<td>READ</td>
<td>The number of segments read.</td>
</tr>
<tr>
<td>DELETED</td>
<td>The number of segments deleted.</td>
</tr>
<tr>
<td>INSERTED</td>
<td>The number of segments inserted.</td>
</tr>
</tbody>
</table>
Twin Segment Statistics

The Twin Segment Statistics report contains information about twin segments. The Unload function and The Reload function generate this report automatically. When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition. The Twin Segment Statistics report provides the following information:

### Table 315  Twin Segment Statistics report fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TWIN NAME</td>
<td>The name of the twin segment.</td>
</tr>
<tr>
<td>NUMBER OF CHAINS</td>
<td>The number of chains for this twin segment.</td>
</tr>
<tr>
<td>MAX CHAIN LENGTH</td>
<td>The maximum chain length for this twin segment.</td>
</tr>
<tr>
<td>AVG CHAIN LENGTH</td>
<td>The average chain length for this twin segment.</td>
</tr>
</tbody>
</table>

#### User Exit Routine Action

The User Exit Routine Action report is generated in the following situations:

- A user exit routine deletes any segments during a database unload (Unload function).
- A user exit routine deletes any segments during a database load or reload (Reload function).
- Twin segment sequence errors are encountered and you specified SEQERROR(SAVE) in the control statement data set (Reload function). Segments written to the sequence error data set are included in the **DELETED BY USER** column of the User Exit Routine Action report. For more information, see “SEQERROR” on page 537.

The User Exit Routine Action report provides the following information:

### Table 316  User Exit Routine Action report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGMENT</td>
<td>The segment’s name.</td>
</tr>
<tr>
<td>LVL</td>
<td>The segment’s level within the database hierarchy.</td>
</tr>
<tr>
<td>PARENT</td>
<td>The segment’s parent segment.</td>
</tr>
<tr>
<td>INPUT COUNT</td>
<td>The number of occurrences of this segment that were read from the input data set.</td>
</tr>
</tbody>
</table>
The Variable Length Segment Statistics report contains information about each variable-length segment. A segment with compression will be variable length in the database even if it appears to the user as a fixed-length segment. The Unload function and the Reload function generate this report automatically. When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition.

The Variable Length Segment Statistics report provides the following information:

### Table 317 Variable Length Segment Statistics report fields (part 1 of 2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGMENT</td>
<td>The segment’s name.</td>
</tr>
<tr>
<td>LENGTHS MIN</td>
<td>The minimum data length specified in the DBD.</td>
</tr>
<tr>
<td>LENGTHS MAX</td>
<td>The maximum data length specified in the DBD.</td>
</tr>
<tr>
<td>LENGTHS PFX</td>
<td>The segment’s prefix length (in bytes).</td>
</tr>
<tr>
<td>LENGTHS DATA</td>
<td>The average data length of the segments found in the database.</td>
</tr>
<tr>
<td>SEGMENT COUNT</td>
<td>The number of segments whose actual data length was less than the MIN value, equal to the MIN value, and greater than the MIN value. These counts are accumulated by comparing the actual segment length against the MIN and MAX values.</td>
</tr>
</tbody>
</table>
The VL Segment Split Statistics report contains split segment information. The Unload function generates this report automatically. When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition. The VL Segment Split Statistics report provides the following information:

**Table 317 Variable Length Segment Statistics report fields (part 2 of 2)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| COMPRESSION FACTOR   | If this segment specifies a user edit/compression routine, the compression factor is expressed as a percentage. The compression factor is the number of bytes before compression minus the number of bytes after compression, divided by the number of bytes before compression: 

\[
\frac{(B - A)}{B}
\]

For example, if the segment was compressed from 100 to 40 bytes this field would show 60 percent. If the segment cannot be compressed any smaller than its original size, you will see a negative compression factor. A compression field up to 10 bytes can be appended to the segment. |
| COMPRESSION NAME      | If this segment specifies a user edit/compression routine, this is the compression routine’s name. |

**VL Segment Split Statistics**

The VL Segment Split Statistics report contains split segment information. The Unload function generates this report automatically. When the database uses the IMS/ESA Partition Support Product, one report is generated to describe the entire database and one report is generated for each partition. The VL Segment Split Statistics report provides the following information:

**Table 318 VL Segment Split Statistics report fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGMENT</td>
<td>The segment name.</td>
</tr>
<tr>
<td>TOTAL NUMBER</td>
<td>The total number of segments found for this segment type. This number is the sum of segments NOT SPLIT and ARE SPLIT.</td>
</tr>
<tr>
<td>NOT SPLIT</td>
<td>The total number of segments not split.</td>
</tr>
<tr>
<td>ARE SPLIT</td>
<td>The total number of split segments for this segment type.</td>
</tr>
<tr>
<td>PFX + DATA IN SAME BLOCK</td>
<td>The total number of split segments whose prefix and data portions remain in the same block.</td>
</tr>
<tr>
<td>PFX + DATA IN DIFF BLOCK</td>
<td>The total number of split segments whose prefix and data portions were forced into different blocks.</td>
</tr>
</tbody>
</table>
Appendix A  Diagnostic procedures

If you experience problems with a MAXM Reorg solution, use the information in this appendix before you call BMC Customer Support. This appendix includes the following topics:

Return codes 617
- Online Reorg function and Reorg function 618
- Unload function 618
- Reload function 619
- Index Build function 619
- Prefix Resolution/Update function 619

Diagnostic DD statements 620
- $BMC$MOD DD 620
- PLUSDUMP DD 620
- PLUSLIST DD 620
- PLUSOUT DD 621
- SYSUDUMP DD 621

Trace 621
- BMCTRACE DD 621
- APITRACE DD 622
- TRACE keyword 622

Online Reorg function problem resolution 622
Using the Trace facility with the Reorg function 623
- Keywords 623
- Coding procedures 624

Return codes

This section describes the return codes, completion codes, and abend codes that MAXM Reorg solutions can issue.

If a function terminates with a user abend, a message is always written to the BMCMSG data set before the utility issues the abend. The user abend code is 4094 or 4095.
Online Reorg function and Reorg function

If any acting utility or function (Unload function, Reload function, Index Build function, Image Copy function, or the prefix resolution and update function in the Reorg function) encounters a problem, Online Reorg function or Reorg function issues a 4094 or 4095 user abend.

The following DBRC abends might be issued:

- If DBRC authorization fails, a U0047 abend (DBRC authorization failed) might be issued.
- If DBRC signon fails, a U0041 abend (DBRC signon failed) might be issued.

Unload function

The Unload function issues the following return codes:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No problems were encountered.</td>
</tr>
<tr>
<td>4</td>
<td>Warning messages were issued. One possible reason this message is issued is that the database was empty.</td>
</tr>
<tr>
<td>8</td>
<td>A function abend occurred. One possible reason this message is issued is that the Unload function encountered pointer errors or sequence errors and PTRERROR(ACCEPT) or SEQERROR(ACCEPT) were specified.</td>
</tr>
<tr>
<td>&gt;8</td>
<td>A severe error occurred; a BMC error message was issued.</td>
</tr>
<tr>
<td>100+</td>
<td>A user exit requested termination.</td>
</tr>
</tbody>
</table>
Reload function

The Reload function issues the following return codes:

**Table 320  Reload function return codes**

<table>
<thead>
<tr>
<th>Return code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No problems were encountered.</td>
</tr>
<tr>
<td>4</td>
<td>Warning messages were issued. The function completed.</td>
</tr>
<tr>
<td>4 - 100</td>
<td>A function terminated abnormally. a BMC error message was issued</td>
</tr>
<tr>
<td>100+</td>
<td>The user exit requested termination.</td>
</tr>
</tbody>
</table>

Index Build function

The Index Build function issues the following return codes:

**Table 321  Index Build function return codes**

<table>
<thead>
<tr>
<th>Return code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No problems were encountered.</td>
</tr>
<tr>
<td>4</td>
<td>Warning messages were issued.</td>
</tr>
<tr>
<td>12</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>16</td>
<td>A function terminated abnormally, an I/O error occurred, a sort terminated</td>
</tr>
<tr>
<td></td>
<td>abnormally, or an error was found in a control statement.</td>
</tr>
<tr>
<td>&gt;16</td>
<td>A BMC error message was issued.</td>
</tr>
</tbody>
</table>

When you initialize an empty index, the Index Build function issues a return code 4.

Prefix Resolution/Update function

The Prefix Resolution/Update function issues return code 0, indicating that no problems were encountered.
Diagnostic DD statements

You can include the following DD statements in your JCL to help with diagnostics:

- $BMC$MOD
- PLUSDUMP
- PLUSLIST
- PLUSOUT
- SYSUDUMP

$BMC$MOD DD

The $BMC$MOD DD statement is used to provide detailed diagnostic information for problem resolution. Include this DD statement only at the request of BMC Customer Support.

The input stream will vary, depending on what BMC Customer Support requests. The module name specified here must begin in column 1. If BMC Customer Support asks you to specify the $BMC$MOD DD statement in JCL, use the following format:

```
//$BMC$MOD DD *
DLIOREC
/*
```

PLUSDUMP DD

The PLUSDUMP DD statement causes the Prefix Resolution/Update function to write a potentially large amount of additional information to the PLUSOUT DD statement. Please include this statement only at the request of BMC Customer Support.

PLUSLIST DD

The PLUSLIST DD statement is used by the Prefix Resolution/Update function to write some reports that might be useful to BMC Customer Support to diagnose problems. The amount of data written to this data set is small. BMC recommends that you use this optional DD statement.
PLUSOUT DD

The PLUSOUT DD statement is used by the Prefix Resolution/Update function to snap the Prefix Resolution/Update function internal control blocks. These internal control blocks are useful to BMC Customer Support for problem determination. If a SYSUDUMP is available, these control blocks can be found in the OS dump.

SYSUDUMP DD

The SYSUDUMP DD statement should be present for most abend conditions. If it is not present, it is dynamically allocated.

NOTE

Abbreviated dumps and Abend-Aid dumps do not provide the information that BMC needs for problem resolution. Always obtain a complete dump. An ABLNDUMP DD DUMMY will be dynamically allocated to cause Abend-Aid to take its abbreviated dump and a full SYSUDUMP.

Trace

BMC Customer Support might ask you to include trace mechanisms in your job. This section explains the trace methods available.

BMCTRACE DD

The BMCTRACE DD statement is used to provide diagnostic information for BMC Customer Support to analyze a problem.

Include a BMCTRACE DD statement in your JCL if you need to analyze the performance of the job. If the overall return code is 8 or higher, a BMCTRACE data set is dynamically allocated. However, if the data set is allocated during the job, the trace might not include records created at the beginning of the job.

If you include the BMCTRACE DD statement in your JCL, specify it as follows:

```//BMCTRACE DD SYSOUT=*```

Have the information from the BMCTRACE data set available when you call BMC Customer Support.
APITRACE DD

The APITRACE DD statement is used by the Reload function API to snap various internal control blocks. These internal control blocks are useful to BMC Customer Support for problem determination. You do not need to include an APITRACE DD statement in the JCL; if an APITRACE data set is needed, it is dynamically allocated. If a SYSUDUMP is available, these control blocks can be found in the OS dump.

TRACE keyword

The TRACE keyword is available through the PLUSIN control statement data set for the Prefix Resolution/Update function only. It causes a trace of the WF1 and WF2 records that are processed through Parent-Child Resolution (PRPURG20) and Logical Twin Resolution (PRPURG30). If you specify TRACE(Y) on the GLBL statement, all work file records for all logical relationships are traced. If you specify TRACE(Y) on the LREL statement, tracing is performed only for the specified logical relationships. TRACE(Y) seriously degrades performance and should be used only for diagnostic purposes.

Online Reorg function problem resolution

The following documentation is required for BMC Customer Support to diagnose and resolve problems with the Online Reorg function:

- contents of BMCMSG data set
- contents of BMCTRACE data set
- job output (JESMSGS, JESLOG, and so on)
- SYSUDUMP (if an abend occurs)
- copy of the IMS log (control region) for the time of the reorganization
- copy of the IMS log (control region) that indicates the Online Reorg function initialization completed successfully
- copy of the IMS log (control region) that indicates any IXC messages

If the job appears to be hanging, BMC might request a CANCEL with a dump.
Contact BMC Customer Support to determine if any additional traces should be performed and if any additional documentation is required.

**Using the Trace facility with the Reorg function**

The Image Copy function offers a Trace facility to assist in problem resolution.

---

**WARNING**

Because this facility produces a large volume of output, BMC recommends that you activate it only when requested by your BMC Customer Support representative.

---

You can use either of two methods to activate Image Copy function tracing:

- Change the job step JCL.
- Use the TRACE control statement.

The Trace facility generates Image Copy Output Records—the output image copy written to the output file. If the data is compressed, the Trace facility prints both the compressed and decompressed forms.

---

**NOTE**

When using the TRACE control statement to activate the Trace facility, diagnostic information is limited to the database data sets that you explicitly code on the TRACE control statement.

---

**Keywords**

The following keywords are required on the TRACE control statement: method.

- DBD
- DDN

The following keywords are optional on the TRACE control statement:

- KEYPSTART
- KEYPSTOP
- START
- STOP
Coding procedures

Select one of the following methods to activate Image Copy function tracing:

- Change the job step JCL.
- Use the TRACE control statement.

**Changing the job step in the JCL**

Complete the following procedures to activate the Trace facility through the job step.

- Code the PLUSOUT DD statement in the execution JCL.
- Add the following DD statement to the job step JCL:

  ```
  //PLUSDUMP DD DUMMY
  ```

  The Image Copy function writes diagnostic information to the PLUSOUT DD statement. If you omit the PLUSOUT DD statement, the Image Copy function dynamically allocates it.

**Using the TRACE control statement**

Complete the following procedures to activate the Trace facility using the TRACE control statement. Keywords on the TRACE control statement identify the database data set or area that is traced and selects the records (by RBN/RBA or by key range) that the Image Copy function prints.

1. Code the DBD and DDN keywords on the TRACE control statement.

2. Code any of the remaining keywords listed in “Keywords” on page 623 to indicate the beginning and ending block or key range.

The following are two examples of how you might use the Trace facility.

Trace all input and output records for the block at RBA 42FC:

```
Trace all input and output records for the blocks containing a key range from ALPHA to BETA:

//ICPSYSIN DD * * KSDS FILE *
IC DBD(DBD02) DDN(DDN01)
TRACE DBD(DBD02) DDN(DDN01) KEYPART(ALPHA) +
    KEYSTOP(BETA)
Post-processing

This appendix describes post-processing tasks used by the Reorg function and the Online Reorg function. This appendix includes the following topics:

- Post-processing
- Logically related databases
- Manual post-processing tasks
- Restart process
- Restart process using the Restart Swap utility
- Restart process using the reorganization job JCL

Post-processing

Post-processing is the time during the Reorg function or the Online Reorg function reorganization after the database has been reorganized but before the database is ready for access by users. Because the database has been reorganized, the post-processing tasks must complete.

Post-processing includes the following tasks:

- data set name swapping
- DBRC notifications
- restart processing, if needed

**NOTE**

If for any reason the job does not complete the post-processing tasks, you must execute the restart process or perform the post-processing tasks manually.

If you specify SWAP(NO) to delay the post-processing tasks, you must execute the restart process to swap the data set names. For more information, see “Restart process” on page 631.
The instructions in this section apply to a standard Reorg function or Online Reorg function reorganization. If you specified INPUT(UNLDFILE) for the Reorg function, you cannot delay post-processing tasks. However, if a failure occurs during post-processing, you can restart the tasks by following the instructions in “Restart process” on page 631.

**NOTE**

Although you can perform the post-processing tasks manually, BMC recommends that you specify options to let the products complete post-processing tasks automatically.

During post-processing, messages are issued that report the beginning and completion of each task; these messages are written to the BMCMMSG data set. Figure 97 shows a sample of post-processing messages written during a Reorg function reorganization that involved a primary database, a primary index, and three secondary indexes:

**Figure 97  Reorg function post-processing messages**

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Message Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC90043I</td>
<td>POST PROCESSING STARTED</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTOMER) NOAUTH</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTINDEX) NOAUTH</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTINDEX1) NOAUTH</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTINDEX2) NOAUTH</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTINDEX3) NOAUTH</td>
</tr>
<tr>
<td>BMC90044I</td>
<td>IDCAMS PROCESSING STARTED FOR FRF/EP DATA SET SWAPPING</td>
</tr>
<tr>
<td>BMC90045I</td>
<td>IDCAMS PROCESSING ENDED WITH RETURN CODE 0</td>
</tr>
<tr>
<td>BMC90080I</td>
<td>NOTIFY.REORG DBD(CUSTOMER) RUNTIME(961591346259)</td>
</tr>
<tr>
<td>BMC90080I</td>
<td>NOTIFY.REORG DBD(CUSTINDEX) RUNTIME(961591346259)</td>
</tr>
<tr>
<td>BMC90080I</td>
<td>NOTIFY.REORG DBD(CUSTINDEX1) RUNTIME(961591346259)</td>
</tr>
<tr>
<td>BMC90080I</td>
<td>NOTIFY.REORG DBD(CUSTINDEX2) RUNTIME(961591346259)</td>
</tr>
<tr>
<td>BMC90080I</td>
<td>NOTIFY.REORG DBD(CUSTINDEX3) RUNTIME(961591346259)</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTOMER) AUTH READOFF</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTINDEX) AUTH READOFF</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTINDEX1) AUTH READOFF</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTINDEX2) AUTH READOFF</td>
</tr>
<tr>
<td>BMC90086I</td>
<td>CHANGE.DB DBD(CUSTINDEX3) AUTH READOFF</td>
</tr>
<tr>
<td>BMC90043I</td>
<td>POST PROCESSING COMPLETED</td>
</tr>
</tbody>
</table>

Figure 98 shows a sample of post-processing messages written during an Online Reorg function reorganization that involved a primary database and a primary index.

**Figure 98  Online Reorg function post-processing messages (part 1 of 2)**

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Message Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC90043I</td>
<td>POST PROCESSING STARTED</td>
</tr>
<tr>
<td>BMC90044I</td>
<td>IDCAMS PROCESSING STARTED FOR FRF/EP DATA SET SWAPPING</td>
</tr>
<tr>
<td>BMC90045I</td>
<td>IDCAMS PROCESSING ENDED WITH RETURN CODE 0</td>
</tr>
<tr>
<td>BMC90080I</td>
<td>NOTIFY.REORG DBD(HIPOQC) RUNTIME(970841121135)</td>
</tr>
<tr>
<td>BMC90080I</td>
<td>NOTIFY.REORG DBD(HIIVQC) RUNTIME(970841121135)</td>
</tr>
<tr>
<td>BMC90083I</td>
<td>NOTIFY.IC DBD(HIPOQC) DDN(HIPOQC) ICDSN(CRF.FAMQC.HIPOQC.HIPOQC.IC1) RUNTIME(970841121139)</td>
</tr>
</tbody>
</table>
In the Reorg function, you can reorganize to shadow data sets or to an image copy, depending on the value of the SWAP keyword. In the Online Reorg function, you always reorganize to shadow data sets.

If you reorganize to shadow data sets (SWAP(YES) or SWAP(DELETE)), the Reorg function or the Online Reorg function can automatically swap data set names and notify DBRC of the reorganization. If you specify SWAP(NO), you must perform these tasks manually.

If you reorganize to an image copy (SWAP(YES,IC) or SWAP(S,IC)), the Reorg function restores the image copy after the reorganization and completes the DBRC notifications. The SWAP(x,IC) option is available in the Reorg function only.

You can use the IDCAMS and DBRC commands provided in this appendix as examples. (For details about DBRC and IDCAMS commands, see the IBM references.) For more information about how the Reorg function and the Online Reorg function work with DBRC, see Appendix C, “DBRC support.”

Logically related databases

If you are reorganizing databases that have logical relationships, restrictions apply. For Online Reorg function limitations, refer to “Online Reorg function” on page 72; for Reorg function limitations, refer to “Reorg function” on page 73.

The multi-step reorganization (see “Multi-step reorganization” on page 88) requires the following post-processing steps:

1. In the reorganization job JCL, specify SWAP(NO) for each database.
2. Provide an AMSOUT data set. Specify it in the JCL, or include an AMSOUT option in the global options module.
3. Execute the reorganization JCL to reorganize each database.
Manual post-processing tasks

The following steps apply when you complete post-processing tasks manually:

**NOTE**
You must perform the steps exactly as listed in this section.

1. Issue a /DBR for each database involved to obtain exclusive control of the database.

2. Create IDCAMS ALTER statements to swap the database data set names. The following is an example of IDCAMS ALTER statements:

   ```sql
   ALTER dbd.dsn NEWNAME ('your.own.temp')
   ```

   VSAM has cluster and component names; these must be renamed together.

3. If DBRC is used, notify DBRC that the reorganization has taken place. Issue the following commands:

   ```sql
   NOTIFY.REORG DBD('YOURDBD') DDN('EACHDDN') CURRENT
   ```

4. If you are using the Reorg function with the BMC DATABASE INTEGRITY PLUS (DI+™) product, you must use the LBLGEN process to rebuild the label for the reorganized database manually. For more information, see the DATABASE INTEGRITY PLUS User Guide.

5. When you have successfully swapped the database data set names, take an image copy of the databases.

6. If DBRC is used, issue the following DBRC command to authorize the database:

   ```sql
   CHANGE.DB DBD('YOURDBD') AUTH
   ```

5. After all databases are reorganized, execute the restart process for each database. The restart process completes the post-processing tasks. See “Restart process” on page 631.

6. Issue the following DBRC command to authorize each database:

   ```sql
   CHANGE.DB DBD('YOURDBD') AUTH
   ```
7 If necessary, issue /STA for the database.

Restart process

Complete the restart process only when either of the following situations occur:

- Post-processing tasks fail.
- You postponed data set name swapping.

Do not use the restart process to restart a reorganization job that failed before post-processing tasks began. If a reorganization job failed before post-processing phase began, you must resubmit the reorganization job with a REORG command in the PLUSIN control statement data set.

The restart process will complete successfully only if the reorganization job had started the post-processing phase. Message BMC90043I POST PROCESSING STARTED is issued when post-processing tasks begin. Before executing the restart process, ensure that message BMC90043I was issued.

You can complete the restart process in either of the following ways:

- Execute the Restart Swap utility (program name FRFREST0). BMC recommends using this method. The AMSOUT data set that was created during the reorganization is required. For more information, see “Restart process using the Restart Swap utility” on page 632.

- Use the reorganization job JCL, and replace the REORG command with a RESTART command. Remove all input/output data sets except for the AMSOUT data set from the reorganization job. The AMSOUT data set that was created during the reorganization is required. For more information, see “Restart process using the reorganization job JCL” on page 633.

**WARNING**

DBRC is required for the restart process. The AMSOUT data set that was created in the reorganization job is required. Do not modify the AMSOUT data set.

When you execute one of the restart processes, all remaining IDCAMS ALTER commands, DBRC notifications, and/or image copy restores are completed. The databases remain in NOAUTH status so that you can verify that all IDCAMS ALTER commands, DBRC notifications, and/or image copy restores have completed successfully. You must reset the NOAUTH flag manually using the DBRC utility.
Restart process using the Restart Swap utility

Perform the following steps to complete the restart processing using the Restart Swap utility:

**NOTE**
The Restart Swap Utility is not a complete functional replacement to the manual steps previously outlined. At this time, the Restart Swap Utility does not perform the Image Copy process after REORG, nor does it AUTHORIZE and START the database. These additional steps should be executed after the completion of the Restart Swap utility. These steps equate to the last three steps outlined in the manual post-processing tasks, previously outlined in this appendix.

1. Execute FRFREST0. Specify the DBD name of the database being reorganized on the PARM parameter of the EXEC statement.

   If the database is a HALDB, the AMSOUT data set is a PDS, and selected partitions were specified. Specify the partition name provided in message BMC18022I following the DBD name, separated by a comma.

2. Include the AMSOUT data set that was created during the reorganization job.

3. Include the same IMS libraries (RECON, RESLIB, IMS, and so on) that were used in the reorganization job. If you made DBD changes, you must also specify the former IMS DBDLIB. If you do not, the restart step fails.

4. Do not include any DD statements for the databases and image copy data sets.

**WARNING**
The Online Reorg function and the Reorg function always require exclusive control during the data set name swapping process. If you attempt a multiple step JCL allocation, exclusive control is lost and the database becomes unavailable.

Figure 99 on page 633 shows sample JCL. This sample is also included in member RESTART of the sample library.

**NOTE**
If you are running restart for a job that included the GROUP subcommand, run only one restart job. You can include the name of any one of the databases from the GROUP subcommand for the PARM keyword, but only one restart job should be run. Restart automatically runs for all of the databases.
Follow these steps to use the RESTART command in place of the REORG command in reorganization JCL:

1. Use the existing JCL that was used for the reorganization (PGM=DFSRRC00,PARM='ULU,DFSURGL0,dbdName') with the following modifications:

   A. Retain the AMSOUT data set that was created during the reorganization job.

   B. Retain the IMS libraries (RECON, RESLIB, IMS, and so on) that were used in the reorganization job. If you made DBD changes, you must specify the former IMS DBDLIB. If you do not, the restart step fails.
**C** Remove all DD statements for the databases and image copy data sets.

---

**WARNING**

The Online Reorg function and the Reorg function always require **exclusive** control during the data set name swapping phase. If you attempt a multiple step JCL allocation, exclusive control is lost and the database becomes unavailable.

---

2 Remove the REORG command and all keywords from the PLUSIN control statement data set. Include the RESTART command in the PLUSIN control statement data set.

If the database is a HALDB, the AMSOUT data set is a PDS, and selected partitions are selected, specify the partition name provided in message BMC180224H in the PART() keyword for the restart command.

**Figure 100** shows sample JCL. This sample is also included in member RESTART1 of the sample library.

**Figure 100  Restart process (RESTART command) sample JCL (part 1 of 2)**

```plaintext
//JOBCLASS JOB (acct),’RESTART POST’.,MSGCLASS=X,MSGLEVEL=(2,0),
//          CLASS=A,NOTIFY=uid
/*JOBPARM K=110
******************************************************************************
//*          REORG FUNCTION RESTART PROCESS USING EXISTING REORG JCL
//*   - DOES THE IDCAMS ALTER(S) & DBRC NOTIFICATION
//*     AND/OR THE IC/RESTORE THAT WAS NOT COMPLETED IN THE REORG
//*   NOTE:  THE PREVIOUS REORG MUST HAVE BEEN IN THE POST-PROCESSING
//*          PHASE FOR THIS STEP TO BE SUCCESSFUL.  IT ALSO MUST USE
//*          THE AMSOUT DATASET CREATED IN THE REORG.
******************************************************************************
//REORGJCL EXEC PGM=DFSRRC00,PARM='ULU,DFSURGLO,dbdName',REGION=0M
//STEPLIB DD  DISP=SHR,DSN=bmc.xxx.load
// DD DISP=SHR,DSN=IMSVS.RESLIB
//DFSRESLB DD DISP=DSH,DSN=IMSVS.RESLIB
//IMS DD DISP=DSH,DSN=IMSVS.DBDLIB
//OLDIMS DD DISP=SHR,DSN=IMSVS.DBDOLD <==if DBD changes
//*------------------------------------------------------------------
//*  REMOVE ALL INPUT/OUTPUT DATASETS EXCEPT FOR AMSOUT
//*------------------------------------------------------------------
//*obdname dd disp=shr,dsn=your.original.database <==remove
//*dbdname dd disp=shr,dsn=your.shadow.database <==remove
//*ic1 dd disp=shr,dsn=your.imageCopy.dataSet <==remove
//*------------------------------------------------------------------
//PLUSIN DD *
RESTART PART(partName) <== replace
REORG command
/*
//AMSOUT DD DISP=SHR,DSN=your.restart.info.dataSet <==required
//BMCMSG DD SYSOUT=*
Figure 100  Restart process (RESTART command) sample JCL (part 2 of 2)

```plaintext
//BMCPRINT DD SYSOUT=*  
//BMCTRACE DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//*
```
Restart process using the reorganization job JCL
This appendix discusses how MAXM Reorg solutions work with DBRC. This appendix includes the following topics:

- DBRC and reorganization
- Force DBRC usage
- RECON data sets
- DBRC and utilities
- Notifications
- DBRC authorization
- Unload function
- Reload function
- Index Build function
- NOTIFY.UIC and recovery
- DBRC and NONRECOV
- Reorg function
- Data set name swapping
- Reorg function checking
- Online Reorg function

DBRC support

DBRC control is essential to the integrity and success of the reorganization process. DBRC manages the access to the database by granting varying degrees of authorization to the database or by prohibiting access. After a reorganization, all recovery resources relating to the disorganized version of the database are obsolete, and using them will damage the database. DBRC must be notified that a reorganization has taken place and when it took place, so that it will not try to use the obsolete resources for recovery. This section describes how the MAXM Reorg solutions work with DBRC during a reorganization.
Force DBRC usage

The Online Reorg function and Online/Defrag function require full DBRC control, meaning that DBRC must be active, and all databases must be registered. BMC recommends that you register all databases and database data sets and that you turn the FORCER flag on. You can accomplish these tasks by using the IMS DBRC utility (DSPURX00).

HALDBs require that DBRC be active.

Force DBRC usage

When you specify DBRC=FORCE during IMS system generation (IMSGEN), you cannot override DBRC usage with commands and keywords in the PLUSIN control statement data set. For example, if DBRC=FORCE was specified during IMSGEN and you specify the Index Build function DBRC(NO) keyword, the Index Build function abends with a U0047 abend.

Specifying DBRC=FORCE in the IMSGEN does not guarantee that DBRC will be implemented fully. When you specify DBRC=FORCE, DBRC checks only to see whether the database is registered. DBRC does not ensure that all database data sets are registered; therefore, you could have data sets that are completely unknown to DBRC that are able to obtain valid authorizations. To ensure that all databases and all data sets are under DBRC control, specify FORCER (force registration) when you initialize the RECON data sets. If you run without FORCER, your data sets do not have the level of protection you probably want. Instead, you can register all databases and database data sets and turn the FORCER flag on. You can accomplish these tasks by using the IMS DBRC utility (DSPURX00).

RECON data sets

DBRC uses the RECON data sets to record information. The RECON data sets are required in every job step that requires DBRC to be active. You can specify the RECON data sets in either of two ways:

- Add a library that contains MDA definitions for the RECON data sets to the STEPLIB concatenation. BMC recommends that you specify RECON data sets in this way. You can also store the MDAs (dynamic allocation members) for the RECON data sets in the RESLIB.

- Specify RECONn DD statements in the JCL.
DBRC and utilities

Traditional utilities cannot use DBRC to prohibit database access for any processes other than those specific to the utility. For example, a reload utility works with DBRC only during the reload process itself, not before the reload begins. This can be dangerous when you are completing an entire reorganization because an update program could obtain access to the database between the unload and the reload. The result would be a loss of data, which might be hard to diagnose. The only way of preventing this unauthorized access is to set the database in NOAUTH mode before the unload and reset it after the reload. With the database in NOAUTH mode, those jobs would receive an authorization error and then abend with a U0047 abend.

Unload jobs must run in an environment that ensures that the database cannot be updated while the data is being unloaded. To ensure that the unload job will complete while no user or program has access to the database, the utility obtains read exclusive authorization. Read exclusive authorization means that the utility can read the database and no other programs or users can update it. If this level of authorization is not obtained, the unload job fails and issues a message stating that improper DBRC authorization was obtained.

Load and reload jobs also must run in an environment that ensures that the database is not used while the data is being reloaded. To protect the database from updates, the utility acquires exclusive authorization. This prevents all programs and users from accessing the database. Because the status of the database data sets is unclear until the load process has completed, the database must be protected from access. This applies to the primary database as well as to the index databases.

Notifications

A reorganization utility issues the NOTIFY.REORG command at the end of the reorganization. DBRC must know that the database has been reorganized and that the image copies and log data sets from before the reorganization cannot be used for recovery (and change accumulation) across this point. To ensure that a valid image copy of the reorganized database will be created (and to give you a way to recover the database if needed), the NOTIFY.REORG command automatically turns the ICNEEDED flag on. No program or user can access the database until the ICNEEDED flag is turned off. You can turn off the ICNEEDED flag in either of the following ways:

- Run an image copy utility. The utility turns off the ICNEEDED flag when an image copy is taken successfully.

- Specify NOTIFY.UIC to tell DBRC that you do not need an image copy for recovery purposes. Use this method only if you have a alternate way of recovering those databases. This is a good method for index databases and the Index Build function, because the Index Build function can recreate the index databases from the primary database.
The authorization process in DBRC works differently during normal application processing and processing performed by utilities. Applications are allowed to run only in clean situations, while some utilities are designed to perform cleanup functions (for example, backout) to recreate a clean situation.

### Application processing

The types of authorizations are bound to the share level (SHARELVL) you have specified for the database. The following share levels are available:

<table>
<thead>
<tr>
<th>Share level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>no sharing</td>
</tr>
<tr>
<td>1</td>
<td>one updater, multiple readers</td>
</tr>
<tr>
<td>2</td>
<td>multiple updaters in single host</td>
</tr>
<tr>
<td>3</td>
<td>multiple updaters in multiple hosts</td>
</tr>
</tbody>
</table>

When you make a DBRC authorization request (through a PSB PROCOPT), you request a specific level of access. The highest level is `exclusive`; the database will not be shared. If the database is already authorized to someone else, your request will be denied. The next authorization level is `update`.

You will not get access if any of the following situations is true:

- SHARELVL(0) and someone has it (with any level of access)
- someone has `exclusive` access
- SHARELVL(1) and someone has update or `read exclusive`
- SHARELVL(2) and another host has update or `read exclusive`

Authorization requests for `read` access will not be honored if either of the following situations is true:

- SHARELVL(0) and someone has it
- someone has `exclusive` access

The special request `read exclusive` will not allow any authorizations higher than `read` at the same time. (`Read exclusive access` is also referred to as `read with integrity`.) Utilities use `read exclusive` access when a database must not be updated during a period of time, for example while an image copy is being taken. For more information, see the discussion about READON/READOFF in the DBRC documentation.
Utilities processing

In some situations, DBRC sets flags to prevent normal processing from being authorized, mainly to force some utilities to run first. The caller must be in recover mode (which most utilities are).

The following special flags are usually seen as roadblocks because they will prevent any application from getting authorized:

- **Image Copy Needed (ICNEEDED)**

  DBRC turns the ICNEEDED flag on after notification of a successful reorganization. No updates are allowed until after an image copy is taken. The completion of the image copy resets this flag.

- **Backout Needed (BACKOUT.NEEDED)**

  The IMS abend exits turn the BACKOUT.NEEDED flag on to enforce a backout if the program performed updates. You can also set this flag if the IMS abend routines cannot be executed (for example, when a power failure has occurred).

- **Recovery Needed (RECOV.NEEDED)**

  The RECOV.NEEDED flag is set when a write error occurs. IMS adds an extend error queue element (EEQE) to the data set entry in DBRC. This EEQE contains the RBA of the block in error. Subsequent calls get an AO status code.

  Authorization works as it normally does. The block in error stays in error. The only way to reset the RECOV.NEEDED flag is to perform a recovery; however, some utilities (such as unload and image copy) will not get authorization because they must see the blocks in error. These utilities require that a recovery be performed. (An initial load can be treated as a recovery.)

- **Prohibit Further Authorization (NOAUTH)**

  The NOAUTH flag prevents any application activity against this database. Utilities can still access the database.

Unload function

The Unload function has no need for a special DBRC interface. Data can be unloaded for reasons other than to be reloaded, so DBRC has no way of knowing whether it needs to protect the data after it is unloaded. You can use DBRC with the Unload function to dynamically allocate the database data sets by using the data set names found in the RECON data sets.
The unload BMP that the Unload function can use under MAXM Reorg/Online does not use any DBRC authorization. The IMS control region handles the authorization for the BMP.

## Reload function

The Reload function and the Index Build function each load data, and they use similar DBRC functions. You can use DBRC with these utilities to dynamically allocate the database data sets by using the data set names found in the RECON data sets. The Reload function and the Index Build function have additional capabilities for using DBRC with the image copy and reorganization notification. A NONRECOV flag signifies that a standard recovery is not required. If you define databases to DBRC as NONRECOV, the Reload function and the Index Build function internally issue a NOTIFY.UIC command after the NOTIFY.REORG command.

The Reload function API issues a NOTIFY.REORG at the end of a user load program. This is not done in any batch application using DL/I batch and a PROCOPT=L PSB. The Reload function considers issuing NOTIFY.REORG an enhancement over other IMS reload utilities and will force you to take an image copy after the user load program has finished.

## Index Build function

BMC recommends the use of DBRC for the Index Build function. This section explains how the Index Build function works with DBRC. You can use DBRC to dynamically allocate the database data sets from DBRC.

The Index Build function can issue a NOTIFY.UIC command. For more information, see “NOTIFY.UIC and recovery” below. You must specify the ICNEEDED keyword as follows to have the Index Build function issue a NOTIFY.UIC command:

- Specify ICNEEDED(ON) to keep the ICNEEDED flag that was set with NOTIFY.REORG on.
- Specify ICNEEDED(OFF) to turn off the ICNEEDED flag with NOTIFY.UIC.

## NOTIFY.UIC and recovery

When you specify ICNEEDED(OFF), the Index Build function issues a NOTIFY.UIC command to inform DBRC that you can recover the database without an image copy.
The Index Build function can recreate all index databases. After you recover the primary database, run the Index Build function to recreate all indexes. The ability to recreate the indexes eliminates the need to save the image copies and logs for the index databases, saving time in the normal backup cycle. It also saves index building time because recreating the indexes is faster than recovering them.

**DBRC and NONRECOV**

DBRC allows you to set databases to be nonrecoverable (NONRECOV). This does not mean that the database cannot be recovered; it means that the normal recovery methods using image copies and logs are no longer viable.

With NONRECOV databases, no after-image records are written to the logs. A backout is possible because the before-image records are still written, but a roll-forward recovery is not possible because the data is no longer there to roll forward with. Therefore, an alternate recovery method is required.

NONRECOV status should never be an option for a primary database. NONRECOV status is an option for indexes because the Index Build function can rebuild indexes quickly.

The Index Build function checks for your DBRC setting (RECOV or NONRECOV) and sets its default accordingly:

<table>
<thead>
<tr>
<th>DBRC setting</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECOV</td>
<td>ICNEEDED(ON)</td>
</tr>
<tr>
<td>NONRECOV</td>
<td>ICNEEDED(OFF)</td>
</tr>
</tbody>
</table>

You can specify that the Index Build function not write any records to the RECON data sets for indexes that are defined as NONRECOV. To do this, specify NONRECOV(N) in the DLIGSET0 module. For more information, see Appendix C, “DBRC support.”

**Reorg function**

BMC recommends the use of DBRC for the Reorg function. This section explains how the Reorg function works with DBRC.

DYNALLOC(YES) and SWAP(YES) require DBRC.
The Reorg function manages a reorganization in one step. Because DBRC authorization is obtained at the beginning of the reorganization and released at the end, no other batch access can be obtained between the unload and the reload.

If DBRC is active and the databases are registered, the MAXM Reorg solution notifies DBRC of the reorganization. The time stamp of the reorganization tells DBRC the time at which image copies and logs recorded in the RECON data sets were made invalid for recovery of the database. After a Reorg function reorganization, all prereorganization data sets can remain intact until the need for a new recovery point (the reorganization) has been recorded in DBRC.

If you executed a standard Reorg function reorganization and you discover any errors before the swap of data set names, abandon the reorganization and delete the output files. If DBRC has not been notified of the reorganization, issue the CHANGE.DB command to allow authorization requests. However, if DBRC has already been notified of the reorganization, the ICNEEDED flag in DBRC might be turned on.

You can delete the reorganization record and the image copy record from the RECON data sets. The DBD is still associated with the data sets it has had all along—the disorganized ones.

### Data set name swapping

When the data set name swap takes place, DBRC notification is critical to guarding the integrity of the database during the events that follow, until the image copy has been taken. The database is most vulnerable between the data set name swap and the creation of a recovery point. Notification of the reorganization does two things:

- Prevents the use of recovery resources created before the time stamp of the reorganization
- Sets a flag (ICNEEDED) to prevent DBRC from allowing update authorization of the database until the image copy is taken. The image copy utility resets the ICNEEDED flag upon successful completion of the image copy.

To eliminate possible database integrity exposure, follow these rules:

- Notify DBRC of the reorganization at the same time you swap the data set names.
- Abandon the reorganization if you are unsure whether the database was updated during the entire reorganization process.

Another way that DBRC protects the database during reorganization is by preventing update access. Updates that take place after the reorganization are at risk of being lost when the data set names are swapped.
If you did not specify automatic data set name swapping, you must execute the restart process to complete the swap and DBRC notification. For more information, see “Restart process” on page 631.

Reorg function checking

The Reorg function requires exclusive or read exclusive access, depending on the value of the SHARE keyword. The Reorg function checks the special flags before it makes any authorization attempts. If it finds certain flags turned on, the Reorg function terminates during initialization (with abend U0047). The special flags are treated as follows:

- **ICNEEDED**

  The Reorg function ignores this flag. When the reorganization completes, an ICNEEDED flag is generated by NOTIFY.REORG.

- **RECOV.NEEDED**

  The Reorg function does not run.

- **BACKOUT.NEEDED**

  The Reorg function does not run.

- **NOAUTH**

  The Reorg function options can be used to override the DBRC condition.

Online Reorg function

The Online Reorg function requires full DBRC registration: DBRC must be active, and all databases and all data sets must be registered. When you reorganize a database with the Online Reorg function, the Reorg function completes its normal DBRC-related tasks. The Online Reorg function performs additional DBRC-related tasks at the beginning of the reorganization and after the change apply process completes. All DBRC activities (and the data set name swapping process) are delayed until the change apply process is finished.
The Online Reorg function creates a file that contains the updates made to the database while it is being reorganized. After the database is reorganized, a batch update program applies these updates to the reorganized database. This batch apply program creates a log, which the Online Reorg function registers to DBRC. To correlate the database to this log, the Online Reorg function creates allocation records.

An image copy was taken of the database as it was reorganized, but the Online Reorg function has not yet issued a NOTIFY.IC. The time stamps in the log records must be after the time stamps in the image copy. For recovery purposes, the Online Reorg function changes the time stamps in the log records to be after the NOTIFY.IC. If you need to recover the database, you will have the correct logs to apply after the image copy.

If you use the image copy that was created during the Online Reorg function reorganization to recover the database, the Online Reorg function batch log should be included. Although the image copy is technically intact (it has no pointer errors), the user data is not complete until the Online Reorg function batch log is applied.

**NOTE**
The Online Reorg function supports an EXIT that enables you to delay the swap and continue logging until a defined period of time. The EXIT module is DLIOWNDW. No keyword is required to activate this feature. If the module is present, the feature is automatically activated. DLIOWNDW is documented in the sample library.
Environment setup member

This appendix discusses how to set installation-wide options for MAXM Reorg solutions. It also explains how to override certain options for a particular job. This appendix includes the following topics:

Overview ................................................................. 647
  Environment setup member ........................................ 648
  BMCSETUP DD ....................................................... 648
  Options .................................................................. 649

Overview

You can set installation-wide options for MAXM Reorg solutions by modifying and assembling the environment setup member (DLIGSET0). You can also override the options created with the environment setup member by specifying a BMCSETUP DD statement in a particular job. This appendix describes the options you can change.

NOTE

The environment setup member does not contain Prefix Resolution/Update function options.

The environment setup member is in #DLIGLBL of the sample library.

Figure 101  #DLIGLBL member (part 1 of 2)

```
//ASM     EXEC ASMHCL
//ASM.SYSLIB DD DISP=SHR,DSN=bmc.xxx.samp
//ASM.SYSIN DD *
SETUP option
  option
END
```
Environment setup member

One of the steps required to install the Online Reorg function and/or the Online/Defrag function is to modify and assemble the environment setup member. The options you specify in this member apply to all of the Online Reorg function and Extended Performance utilities jobs. To override an option for a specific job, use the BMCSETUP DD statement.

**NOTE**

If you have assembled your own installation default module (DLIGSET0), you must reassemble it each time you install a new tape.

**BMCSETUP DD**

You can override installation-wide options for a particular job by specifying the BMCSETUP DD statement. The BMCSETUP DD statement describes a control statement data set that allows you to override the setup options that were set during installation.

---

**Figure 101 #DLIGLBL member (part 2 of 2)**

```plaintext
//LKD.SYSIN DD *
NAME DLIGSET0(R)
//LKD.SYSLMOD DD DISP=SHR, DSN=bmc.xxx.load
```

Change the following information when you assemble the environment setup member:

- `bmc.xxx.samp`
  
  Change this value to reflect your data set.

- `bmc.xxx.load`
  
  Change this value to reflect your load library name.

You can change all of the other options listed. The following section explains the options, lists valid and default values, and explains whether you can change the option in the environment setup member, a particular job, or both.
Use the syntax described in “PLUSIN control statement syntax” on page 404. For example, to specify a date format, code the following in the BMCSETUP control statement data set:

```
DATEFORM(MDY)
```

## Options

Table 324 describes the options you can set for the environment setup member and the options you can override with the BMCSETUP DD statement. The SETUP macro contains more options than are documented here; use the options not documented only at the request of BMC Customer Support.

Unless otherwise noted, these options do not have corresponding PLUSIN keywords.

### Table 324 Environment setup member options (part 1 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGN</td>
<td>agnName</td>
<td>n/a</td>
<td>none</td>
<td>Specify an application group name for the BMP.</td>
</tr>
<tr>
<td>AOI</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Specify whether to use the Online Reorg function automated operator interface.</td>
</tr>
<tr>
<td>AOIDISPC</td>
<td>0-255</td>
<td>AOIDISPC(n)</td>
<td>30 (per try)</td>
<td>Specify the maximum number of times the Online Reorg function will check that a DBR or DBD command worked, before Online Reorg function resends or abandons the command.</td>
</tr>
<tr>
<td>AOITRY</td>
<td>0-99</td>
<td>AOITRY(n)</td>
<td>2</td>
<td>Specify the maximum number of times Online Reorg function will attempt a DBR or DBD command.</td>
</tr>
<tr>
<td>AOIUSER</td>
<td>progName</td>
<td>n/a</td>
<td>none</td>
<td>Specify whether to invoke a user exit routine before issuing a /DBR and after issuing a /STA.</td>
</tr>
<tr>
<td>AOIWAIT</td>
<td>0-99</td>
<td>AOIWAIT(n)</td>
<td>5</td>
<td>Specify the number of seconds that Online Reorg function waits after DBR and DBD commands are sent to IMS and between checks to see if the command worked.</td>
</tr>
</tbody>
</table>
### Options

#### Table 324  Environment setup member options (part 2 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTHREST</td>
<td>A</td>
<td>AUTHREST(N)</td>
<td>N</td>
<td>Specify the HFR RESTART DBRC AUTH/NOAUTH option. The following options are available when run with SWAP(N):</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td>- A:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- Skip the NOAUTH call at the end of the REORG step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- Perform the NOAUTH call at the beginning of the RESTART step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- Check for updates to the DB between the REORG and RESTART steps.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- Turn off NOAUTH at the end of the RESTART step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- N:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- Perform the NOAUTH call at the end of the REORG step.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- Leave the NOAUTH in effect at the end of the RESTART step.</td>
</tr>
<tr>
<td>BMCPAU</td>
<td>0</td>
<td>BMCPAU(0)</td>
<td>0</td>
<td>Specify whether to use the internal pause command in the CRF component:</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>BMCPAU(1)</td>
<td></td>
<td>- 0—Use internal pause.</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>BMCPAU(2)</td>
<td></td>
<td>- 1—Use DBR.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- 2—Use pause, then switch to DBR if unable to find a quiesce point.</td>
</tr>
<tr>
<td>BMPDISPC</td>
<td>0-255</td>
<td>BMPDISPC(n)</td>
<td>30 (per try)</td>
<td>Specify the maximum number of times the Online Reorg function checks that all BMPs that were requested to stop were stopped.</td>
</tr>
<tr>
<td>BMPTRY</td>
<td>0-99</td>
<td>BMPTRY(n)</td>
<td>2</td>
<td>Specify the maximum number of times Online Reorg function asks for BMPs to be stopped before attempting the DBR or before stopping with a WTOR CANCEL or RETRY.</td>
</tr>
<tr>
<td>Option</td>
<td>DLIGSET0 values</td>
<td>BMCSETUP DD values</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------</td>
<td>-------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>BMPWAIT</td>
<td>0-99</td>
<td>BMPWAIT((n))</td>
<td>5</td>
<td>Specify the number of seconds that Online Reorg function waits after requesting that a BMP be stopped and between checks to see if that BMP was stopped.</td>
</tr>
<tr>
<td>CENTURY</td>
<td>Y</td>
<td>CENTURY(Y)</td>
<td>Y</td>
<td>Turn the century indicator on or off.</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>CENTURY(N)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Y—display a 4-digit year (2008)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• N—display a 2-digit year (08)</td>
</tr>
<tr>
<td>CKUPDATE</td>
<td>(Y,STOP4094)</td>
<td>n/a</td>
<td>Y,STOP4094</td>
<td>Specify whether to check the RECON data set for update allocation records dated after the unload file was created. If such records exist, specify the action to take.</td>
</tr>
<tr>
<td>COPYIO</td>
<td>0</td>
<td></td>
<td>0</td>
<td>Specify the behavior of the COPYIO engine. You can specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>• 0—run the data sets in parallel tasks</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• 1—use a single thread copy</td>
</tr>
<tr>
<td>COPYMIN</td>
<td>n/a</td>
<td></td>
<td>1000</td>
<td>Specify the minimum size of the data set, in cylinders, used for a parallel copy.</td>
</tr>
<tr>
<td>CREORG</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td>Specify the conditional reorg option:</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td>• Y—invoke conditional reorg</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• N—execute reorg unconditionally</td>
</tr>
<tr>
<td>CRFUNSP</td>
<td>1-999</td>
<td>CRFUNSP(1-999)</td>
<td>100</td>
<td>Specify the number of cylinders to be used for each extent of the temporary HD Unload file used in the Online Reorg function when no BMP is used. Because this might be a large amount of space, you can use this value to control SMS pool usage.</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>n/a</td>
<td>any valid data class</td>
<td>none</td>
<td>Specify the SMS data class.</td>
</tr>
<tr>
<td>DATE</td>
<td>(MDY,/)</td>
<td>n/a</td>
<td>MDY,/</td>
<td>Specify the date format and separator. Any nonblank character can be the separator. (MDY,/) provides a date format of MM/DD/YY.</td>
</tr>
</tbody>
</table>
### Table 324 Environment setup member options (part 4 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATEFORM</td>
<td>n/a</td>
<td>DATEFORM(MDY)</td>
<td>MDY</td>
<td>Specify the date format.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATEFORM(YMD)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DATEFORM(DMY)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATESEP</td>
<td>n/a</td>
<td>DATESEP(value)</td>
<td>/</td>
<td>Specify the date separator. Any character is valid.</td>
</tr>
<tr>
<td>DBR</td>
<td>F</td>
<td>n/a</td>
<td>F</td>
<td>Specify whether to switch the OLDS during the /DBR at the end of the reorganization.</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFSDBUX1</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Specify if UNLOADPLUS/EP for IMS OR LOADPLUS/EP for IMS should invoke DFSDBUX1 when it is present in the STEPLIB or DFSRESLB:</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td>- Y—invoke DFSDBUX1 when it is present</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- N—do not invoke DFSDBUX1 even when it is present</td>
</tr>
<tr>
<td>DYNALLOC</td>
<td>(Y)</td>
<td>n/a</td>
<td>N</td>
<td>Specify whether to dynamically allocate database data set, output (shadow) database data sets, and the data set name extension for dynamically allocated output data sets.</td>
</tr>
<tr>
<td></td>
<td>(N)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(Y,Y,Z)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(N,Y)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EOJACCES</td>
<td>EX</td>
<td></td>
<td></td>
<td>Specify the access parameter that is appended to the final /STA command issued by a REORG job.</td>
</tr>
<tr>
<td></td>
<td>UP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RO.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXSRTACT</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td>Specify if the ACCT= PARM is to be added to generated IEESYSAS JCL used for the external SECONDARY INDEX UTILITY sort address space. The account data from the batch job is copied to the ACCT= parameter.</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTSORT</td>
<td>SIU</td>
<td></td>
<td></td>
<td>Specify the products that will use the external sort facility:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- SIU—The SECONDARY INDEX UTILITY will use the external sort facility.</td>
</tr>
<tr>
<td>FRFPRINT</td>
<td></td>
<td>BMCPRINT</td>
<td></td>
<td>See “xxxPRINT” on page 522.</td>
</tr>
</tbody>
</table>
Table 324  Environment setup member options  (part 5 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSGNAME</td>
<td>gsgName</td>
<td>n/a</td>
<td>none</td>
<td>Specify a GSG name for RSR.</td>
</tr>
<tr>
<td>HULABSP</td>
<td>1-999</td>
<td>n/a</td>
<td>20</td>
<td>Define the maximum amount of database buffer space to be used for the Unload function Application Program Interface (API).</td>
</tr>
<tr>
<td>HULDBSP</td>
<td>1-999</td>
<td>n/a</td>
<td>20</td>
<td>Define the maximum amount of database buffer space to be used for the Unload function Database Selection Facility (DBSF)</td>
</tr>
<tr>
<td>HULSBSP</td>
<td>1-999</td>
<td>n/a</td>
<td>20</td>
<td>Define the maximum amount of database buffer space to be used for DBSCAN.</td>
</tr>
<tr>
<td>HULUBSP</td>
<td>1-999</td>
<td>n/a</td>
<td>20</td>
<td>Define the maximum amount of database buffer space for the Unload function to use.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>You can override the default you set here by specifying the BUFSACE keyword. See “BUFSACE” on page 413.</td>
</tr>
<tr>
<td>ICP</td>
<td>Y</td>
<td>n/a</td>
<td>N</td>
<td>Specify whether to invoke the Image Copy function.</td>
</tr>
<tr>
<td>IDCAMS</td>
<td>(input)</td>
<td>n/a</td>
<td>„N</td>
<td>Specify input or input and output files for IDCAMS processing and whether to reclaim space after the Online Reorg.</td>
</tr>
<tr>
<td>IMSID</td>
<td>imsid</td>
<td>n/a</td>
<td>none</td>
<td>Use “imsid” as the IMSID portion of the PDX key.</td>
</tr>
<tr>
<td>LANGUAGE</td>
<td>ENGLISH</td>
<td></td>
<td>ENGLISH</td>
<td>Do not change.</td>
</tr>
</tbody>
</table>
Specify how to perform the NOTIFY.REORG LOAD API:

- **F**—For the LOAD API JOBSTEP, perform the NOTIFY.REORG during the step initialization. The F option is more efficient than the B option, but when F is specified, the NOTIFY.REORG is loaded for failed job steps.
- **B**—For the LOAD API JOBSTEP, perform the NOTIFY.REORG during step termination. The B option is less efficient than the F option, but the NOTIFY.REORG is skipped for failed job steps.
- **N**—Do not perform the NOTIFY.REORG LOAD API.

The load data space limit is the maximum space that will be used to create a single database record (for example, a root and all of its children). Sometimes this limit is implemented through MVS parameters.

This limit does not include the data space that might be used to hold segments in memory due to the use of SEGPOS or SEQERROR.

Specify whether to divide data into ranges when unloading it.

Specify a data set to record changed made to the database during the reorganization.
Table 324  Environment setup member options (part 7 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGRETPD</td>
<td>1-99999</td>
<td>n/a</td>
<td>none</td>
<td>Specify how long to keep logs (in days).</td>
</tr>
<tr>
<td>LOGSMS</td>
<td>(Y,mgmt,strg,data)</td>
<td>n/a</td>
<td>none</td>
<td>Use the specified SMS management class, storage class, and data class to allocate the log.</td>
</tr>
<tr>
<td>LOGSPACE</td>
<td>(prim,sec)</td>
<td>n/a</td>
<td>100,50</td>
<td>Allocate the primary and secondary space for the logs in cylinders in the specified amounts.</td>
</tr>
<tr>
<td>LOGUNIT</td>
<td>(unitName, unitName)</td>
<td>n/a</td>
<td>values specified in installation parameters</td>
<td>Use the specified unit name to allocate the primary and secondary logs.</td>
</tr>
<tr>
<td>MAXSORT</td>
<td></td>
<td></td>
<td>3,0</td>
<td>Specify the maximum sort tasks (internal and external) in the following format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MAXSORT (INT,EXT)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>– INT is the maximum number of concurrent internal sorts for a command (BUILD, REORG).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>– EXT is the maximum number of external sorts. Specify 0 for no limit.</td>
</tr>
<tr>
<td>MINDB</td>
<td></td>
<td></td>
<td>2000</td>
<td>Specify the minimum size (in cylinders) of a database that will use parallel processing.</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>n/a</td>
<td>any valid management class</td>
<td>none</td>
<td>Specify the SMS management class.</td>
</tr>
<tr>
<td>MODLDSCB</td>
<td></td>
<td></td>
<td>SYS1.MODEL</td>
<td>Specify the standard model DSCB for GDGS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>The model is used when CRGS logs are dynamically allocated GDGS.</td>
</tr>
<tr>
<td>MONITOR</td>
<td>nnnnnnn</td>
<td>n/a</td>
<td>0</td>
<td>Monitor job progress at specified interval.</td>
</tr>
</tbody>
</table>
### Table 324 Environment setup member options (part 8 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSGWTO</td>
<td>A</td>
<td></td>
<td>n/a</td>
<td>Use this option to force all messages of the specified severities to be written to the operator console and the BMCMSG DD. The last character of the BMC message is the severity level of the message:</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td>- A—an error condition on which the operator must take some action</td>
</tr>
<tr>
<td></td>
<td>E</td>
<td></td>
<td></td>
<td>- C—an error message that results in cancellation of a job</td>
</tr>
<tr>
<td></td>
<td>I</td>
<td></td>
<td></td>
<td>- E—an error message that is the highest in severity</td>
</tr>
<tr>
<td></td>
<td>P</td>
<td></td>
<td></td>
<td>- I—an informational message that is the lowest in severity</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td>- P—an error condition that could adversely affect performance of the product.</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td></td>
<td></td>
<td>- R—an error condition for which the operator must reply</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td>- S—an error message that is the highest in severity</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- W—a warning message that signals that an abnormal condition or problem has been detected</td>
</tr>
</tbody>
</table>
### Table 324  Environment setup member options (part 9 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MVSCMND</td>
<td></td>
<td></td>
<td>Y</td>
<td>Specify whether to process MVS MODIFY and STOP commands:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- Y—allow MVS MODIFY and STOP commands entered at the MVS console. See “MONITOR, MONUSERS, ROUTCODE, DESCCLASS” on page 490.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- N—do not allow MVS console commands. This might be useful if user exits or LOAD/UNLOAD API programs are coded to use the MVS command interface because only one program can be waiting on the command interface at a time. The monitoring set up through keywords will still function if MVSCMND is N.</td>
</tr>
<tr>
<td>NONRECOV</td>
<td>U</td>
<td>NONRECOV(U)</td>
<td>U</td>
<td>Specify whether the Extended Performance Utilities will complete DBRC notifications automatically when the database is registered as NONRECOV.</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>NONRECOV(N)</td>
<td></td>
<td>- U—The product issues a NOTIFY.REORG and a NOTIFY.UIC to DBRC.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- N—The product does not issue any DBRC notifications. Because the NOTIFY.REORG is not issued, the ICNEEDED flag is not set. The NOTIFY.UIC is not issued.</td>
</tr>
<tr>
<td>OSAMMAX</td>
<td>4</td>
<td>OSAMMAX(4)</td>
<td>4</td>
<td>Define the maximum OSAM database size.</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>OSAMMAX(8)</td>
<td>8</td>
<td>You can specify this option only if you are using IMS Version 5.1 or later.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>You can override the default you set here by specifying the OSAMMAX keyword in a PLUSIN control statement data set. See “OSAMINIT” on page 499.</td>
</tr>
<tr>
<td>Option</td>
<td>DLIGSET0 values</td>
<td>BMCSETUP DD values</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------</td>
<td>--------------------</td>
<td>--------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>Y</td>
<td>n/a</td>
<td>Y</td>
<td>Specify whether to process files in parallel mode.</td>
</tr>
<tr>
<td>PCP_FREQ</td>
<td>Y</td>
<td></td>
<td>N</td>
<td>Specify whether POINTER CHECKER PLUS/EP frequency analysis is enabled.</td>
</tr>
<tr>
<td>PCP_THRS</td>
<td></td>
<td>0</td>
<td></td>
<td>Specify the threshold percentage for the frequency report.</td>
</tr>
<tr>
<td>PDXHP</td>
<td>Y</td>
<td></td>
<td>N</td>
<td>Specify whether you want POINTER CHECKER PLUS/EP to generate and update HP type PDX records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>BMC recommends that you move away from using HP PDX records because of the high overhead involved in opening and closing the PDX and reading and writing the PDX directory for each type of PDX record.</td>
</tr>
<tr>
<td>PERMUNIT</td>
<td>any valid unit name</td>
<td>any valid unit name</td>
<td>SYSALLDA</td>
<td>Assign a unit name for permanent data sets.</td>
</tr>
<tr>
<td>PRINT</td>
<td>Y</td>
<td>PRINT(Y)</td>
<td>N</td>
<td>Specify whether to list setup options in job output.</td>
</tr>
<tr>
<td>xxxPRINT</td>
<td>with xxx as FRF, LDP, SIU, and ULP</td>
<td>ddname</td>
<td>n/a</td>
<td>BMCPRINT Use the specified data set to contain reports and other output.</td>
</tr>
<tr>
<td>RECONRDR</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td>Specify the RECON reader option:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ N—Disallow the use of the RECON reader. If you specify N, DBRC must be available to continue.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ Y—Allow the RECON reader to read the RECONs to retrieve information if DBRC is not active.</td>
</tr>
<tr>
<td>RMBYTES</td>
<td>S</td>
<td>n/a</td>
<td>M</td>
<td>Specify whether to use the segment’s VL SIZE field or its MINBYTES value to accumulate database record bytes.</td>
</tr>
</tbody>
</table>
### Options

#### SMS (N)

 specifying whether to supply SMS parms for temporary data sets. In DLIGSET0, also specify management class, storage class, and data class.

There is no corresponding PLUSIN keyword. However, if you specify SMS constructs here, you can override the sort-related SMS data sets with the SORTWORK keyword. See “SORTWORK” on page 545.

#### SNAPERR A

 specifying the SNAPERR keyword to instruct the API about how to react to errors when XBM is being used.

Specify BYPASS or ABEND. If you specify BYPASS, the API bypasses errors as it does when XBM is not being used. If you specify ABEND, the job step will end with a 4094 abend. The default value is ABEND.

#### SNAPSHOT Y

 specifying the software snapshot option:

- N — Read snapshot copy into the Copy function. This holds the snapshot copy for the shortest period.
- Y — Skip the copy process and read the snapshot copy into UNLOAD. This causes a software snapshot to function like the hardware snapshot but it also increases the possibility for abends due to exceeding snapshot buffers.

#### SORTPRM ddname,ddname

 define the ddnames used by your sort product for sort commands.

#### SORTWORK (n,ccc,unitName, dataclas,mgmtclas, storclas)

 dynamically allocate sort work data sets.
Table 324  Environment setup member options  (part 12 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPILL</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Specify SPILL option preference:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ N—The LOAD process builds database records in memory or data spaces.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ Y—SPILL large records to DASD. Useful for databases that have extremely large records.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>See “SPILL” on page 547.</td>
</tr>
<tr>
<td>SSID</td>
<td></td>
<td></td>
<td></td>
<td>See “SSID” on page 547.</td>
</tr>
<tr>
<td>STAOPEN</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Specify whether you want to use the OPEN parameter on the /STA command:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ Y—Add OPEN to /STA command. Causes immediate open of data sets in the IMS control region.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ N—Do not add OPEN to /STA command. Open is delayed until first access.</td>
</tr>
<tr>
<td>STCJOBNM</td>
<td>1-7 characters that will be used as the prefix for the started task name</td>
<td>JOBNAME</td>
<td></td>
<td>Specify the Job name prefix used when creating address spaces. See “EXTSORT” on page 652.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JOBNAME</td>
<td></td>
<td>Specify JOBNAME to force the use of the job name from the originating job.</td>
</tr>
<tr>
<td>STRGCLAS</td>
<td>n/a</td>
<td>any valid storage class</td>
<td>none</td>
<td>Specify the SMS storage class.</td>
</tr>
<tr>
<td>SUBSYS</td>
<td>as many as four DBUSS IDs</td>
<td>one DBUSS ID</td>
<td>DBUZ</td>
<td>Specify a DBUSS name, if using. The DBUSS is used for APF-authorization only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DBUI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYSOUT</td>
<td>*</td>
<td>SYSOOUT(*)</td>
<td>*</td>
<td>Set a default SYSOUT class.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>There is no corresponding PLUSIN keyword.</td>
</tr>
<tr>
<td>TEMPIC</td>
<td>0</td>
<td>TEMPIC(0)</td>
<td>0</td>
<td>Specify whether to allow temporary data sets for image copy data sets:</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>TEMPIC(1)</td>
<td></td>
<td>■ 0—Do not allow temporary data sets for image copies.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ 1—Allow temporary data set for image copies.</td>
</tr>
</tbody>
</table>
### Table 324 Environment setup member options (part 13 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEMPUNIT</td>
<td>any valid unit name</td>
<td>any valid unit name</td>
<td>SYSALLDA</td>
<td>Assign a unit name for temporary data sets.</td>
</tr>
<tr>
<td>TIME</td>
<td>(24,,:) (12,,:)</td>
<td>n/a</td>
<td>24, :)</td>
<td>Display the time in 24-hour format or 12-hour format, and use a colon to separate hours and minutes. In 24-hour format, 8:00 PM displays as 20:00.</td>
</tr>
<tr>
<td>TIMEFORM</td>
<td>n/a</td>
<td>TIMEFORM(24) TIMEFORM(12)</td>
<td>24</td>
<td>Display the time in 24-hour format or 12-hour format. In 24-hour format, 8:00 PM displays as 20:00.</td>
</tr>
<tr>
<td>TIMESEP</td>
<td>n/a</td>
<td>TIMESEP(;)</td>
<td>:</td>
<td>Specify the character to separate hours and minutes. Any nonblank character is valid.</td>
</tr>
<tr>
<td>TRACE</td>
<td>0</td>
<td></td>
<td>0</td>
<td>Specify the FORCE BMCTRACE option:</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>■ 0—Do not allocate a BMCTRACE data set.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ 1—Dynamically allocate a BMCTRACE data set if the DD is not present in the JCL.</td>
</tr>
<tr>
<td>TRACELIM</td>
<td>n/a</td>
<td></td>
<td>n/a</td>
<td>Specify the TRACE output limit in the following format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TRACELIM (ONLINE,BATCH)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ ONLINE—The limit for the IMS online region.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ BATCH—The limit for the batch jobs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>The values are multiplied by 1000.</td>
</tr>
<tr>
<td>TYPE</td>
<td>CSECT</td>
<td>CSECT</td>
<td></td>
<td>Specify the type:</td>
</tr>
<tr>
<td></td>
<td>DSECT</td>
<td></td>
<td></td>
<td>■ CSECT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ DSECT</td>
</tr>
<tr>
<td>UCBITMAP</td>
<td>NO</td>
<td></td>
<td></td>
<td>See “UCBITMAP” on page 557.</td>
</tr>
<tr>
<td>ULPPRINT</td>
<td>BMCPRINT</td>
<td></td>
<td></td>
<td>See “xxxPRINT” on page 522.</td>
</tr>
<tr>
<td>UNAUTH</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td>Specify the DBRC authorization option for UNLOAD:</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td>■ Y—Get DBRC authorization before UNLOAD.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>■ N—Skip DBRC authorization before UNLOAD.</td>
</tr>
</tbody>
</table>
## Table 324  Environment setup member options  (part 14 of 14)

<table>
<thead>
<tr>
<th>Option</th>
<th>DLIGSET0 values</th>
<th>BMCSETUP DD values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XBMSSID</td>
<td>n/a</td>
<td></td>
<td>n/a</td>
<td>Specify the XBM SNAPSHOT subsystem ID.</td>
</tr>
<tr>
<td>XIMGROUP</td>
<td>n/a</td>
<td></td>
<td>DBUXIM00</td>
<td>Specify the XIM group name.</td>
</tr>
</tbody>
</table>
This appendix discusses how you can use an user exit during a reorganization. This appendix includes the following topics:

Overview ................................................................. 663
Sample user exit routines ............................................ 664
Unload function .......................................................... 665
  User exit register usage .............................................. 666
  Return codes .......................................................... 667
Reload function .......................................................... 668
  User exit routine register usage ................................. 669
  Return codes .......................................................... 670
Reorg function .......................................................... 671
Online Reorg function .................................................. 671
  Restrictions ............................................................ 672
  Application considerations ......................................... 672
  How the Online Reorg function processes user exits ....... 673
  User exit routine usage .............................................. 675
  Return codes .......................................................... 676
COBOL exit routines ................................................... 677
  OS/VS COBOL .......................................................... 677
  COBOL II ............................................................... 677
  COBOL/370, COBOL for MVS, and COBOL for OS/390 .... 678
Using the image copy Data Set Name user exit ................. 678
  Coding procedures .................................................. 679
  Assemble and link Data Set Name user exit .................... 686

Overview

The Unload function, the Reload function, the Reorg function, and the Online Reorg function support a user exit routine that allows you to examine, modify, and delete segments.
The following restrictions apply to the use of a user exit:

- If you specify the USEREXIT keyword, databases are unloaded, reloaded, or reorganized in non-parallel mode.

- If you are removing a segment type from a DBD, you must delete all occurrences of the segment data from the HD Unload file before you reload the file with the Reload function or the Reorg function. You can use an Unload function user exit to delete the segment data.

- If you are using a user exit to change a key, execute that user exit when the database is unloaded. In the Reload function, if you use the HD Sort utility, the user exit is invoked after the HD Sort utility. Reload the database, and specify HDSORT(YES).

- User exits in an Online Reorg function job are valid only for HD databases.

**NOTE**

Do not include a data capture exit in a DBD that is used with a user exit. Doing so can create a data integrity exposure because the user exit could modify data outside of the knowledge of the data capture exit.

If you are implementing or changing a DFSDBUX1 exit between the unload and the reload and if the DFSDBUX1 exit will change the root key, you must specify HDSORT(YES) or run the HD Sort utility between the unload and the reload. For more information, see “DFSDBUX1” on page 424.

When you dynamically allocate an output image copy data set, you can let the Image Copy function determine the data set name, or you can provide a Data Set Name user exit routine to supply the data set name.

### Sample user exit routines

Sample user exit routines are members in the sample library:

<table>
<thead>
<tr>
<th>Exit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRFUXASM</td>
<td>sample Assembler user exit routine for the Online Reorg function.</td>
</tr>
<tr>
<td>CRFUXCOB</td>
<td>sample COBOL user exit routine for the Online Reorg function.</td>
</tr>
<tr>
<td>HLDUXASM</td>
<td>sample Assembler user exit routine for the Reload function</td>
</tr>
<tr>
<td>HLDUXCOB</td>
<td>sample COBOL user exit routine for the Reload function</td>
</tr>
<tr>
<td>HLDUXPLI</td>
<td>sample PLI user exit routine for the Reload function</td>
</tr>
</tbody>
</table>
Unload function

The Unload function invokes the user exit routine for each segment before it places the segment in the HD Unload data set (DFSURGU1). The exit is invoked one more time at termination. The following considerations apply to invoking a user exit routine with the Unload function:

- Parallel processing is disabled when you specify USEREXIT.

- If the data is compressed in the input file and you specify that you want to expand compressed segments with the EXPAND keyword, the Unload function automatically expands the data before passing it to the exit. If you specify that you do not want to expand compressed segments, the data is passed to the exit in compressed format.

- When you are using the user exit for fixed-length segments with compression, the following rules apply:
  - If the data is expanded (compression is invoked), do not specify the length field in the work area of the user exit linkage section.
  - If the data is compressed (compression is not invoked), specify the length field in the work area of the user exit linkage section.

- If the segment data is to be used without change, set the return code to 00. When modifying a segment, copy the segment data from A(segment data) to A(work area), change the data at A(work area), and set the return code to 04.

- When deleting a segment, you must delete all hierarchically dependent segments or set the return code to 12.

- The Unload function unloads the segments and passes them to the user exit routine in hierarchical sequence. If your exit program changes the value of the key fields, you must sort the segments into hierarchical sequence before reloading them.

- The user exit routine must be in a library available to the unload job step because the Unload function dynamically loads it. The work area for the user exit must be the same size as the largest segment in the DBD.

<table>
<thead>
<tr>
<th>Exit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HULUXASM</td>
<td>sample Assembler user exit routine for the Unload function</td>
</tr>
<tr>
<td>HULUXCOB</td>
<td>sample COBOL user exit routine for the Unload function</td>
</tr>
<tr>
<td>HULUXPLI</td>
<td>sample PLI user exit routine for the Unload function</td>
</tr>
</tbody>
</table>

Table 325  Sample user exit routines (part 2 of 2)
If the exit was not linked as AMODE=31 and the database buffers are located above the 16 MB line, the Unload function copies the segment to a work area located below the 16 MB line before passing control to the exit routine in AMODE=24. For more information, see “Buffers” on page 186.

If you are using the user exit to copy storage beyond the end of the segment data, you can specify YES as the third subparameter on the USEREXIT option, as follows:

\[
\text{USEREXIT}=(\text{exitName}, \text{YES})
\]

Specifying YES copies the segment to a 32-KB work area before invoking the user exit routine. A slight CPU usage overhead is associated with copying all segments to another area.

**NOTE**

If you specify NO or a blank for the third subparameter, the segment passed to the user exit remains in its original location. If the user exit tries to address data beyond the end of the segment, you might encounter a system S0C4 abend.

**User exit register usage**

Figure 102 shows the requirements for register 1 usage on entry to a user exit routine.

**Figure 102  Unload function user exit routine**

<table>
<thead>
<tr>
<th>Register 1</th>
<th>A(segname)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A(segment data)</td>
</tr>
<tr>
<td></td>
<td>A(work area)</td>
</tr>
<tr>
<td></td>
<td>A(PCB area)</td>
</tr>
<tr>
<td></td>
<td>A(LPCK area)</td>
</tr>
</tbody>
</table>
Table 326  Register 1 requirements for Unload function user exit routine

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A(segname)</td>
<td>Address of a 12-byte area that contains an 8-byte segment name, a 1-byte segment code, a 1-byte segment level code, and a 2-byte (halfword binary) segment data length. At termination, the segment name will contain blanks and the segment code will contain a binary zero.</td>
</tr>
<tr>
<td>A(segment data)</td>
<td>Pointer to the segment data about to be written to the unload file. If the EXPAND(NO) keyword is in effect, this data is in compressed format.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Because the segment is physically located in an I/O buffer, the segment length must not be changed in this area.</td>
</tr>
<tr>
<td>A(work area)</td>
<td>Pointer to an Unload function segment work area large enough to hold the largest possible segment defined in the DBD being used. If a segment length change is required, this area must be used. Because the amount of segment data placed in the HD Unload data set is controlled by the segment size in the DBD, segments can be expanded only if they are defined in the DBD as variable length. The segment size must not be larger than the size of the largest segment in the DBD. Segments defined as fixed length in the DBD can be made longer by the user exit by specifying the NEWDBD keyword. See “NEWDBD” on page 494. Fixed-length segments can be made longer with a similar user exit routine during reload with the Reload function. <strong>Warning:</strong> When moving the segment data from one area to another, ensure that you move only the length of the current segment.</td>
</tr>
<tr>
<td>A(PCB area)</td>
<td>Address of the area containing the PCB information.</td>
</tr>
<tr>
<td>A(LPCK area)</td>
<td>Address of the area containing the LPCK.</td>
</tr>
</tbody>
</table>

**Return codes**

When you exit the user exit routine, register 15 must contain a decimal return code for the Unload function. Table 327 on page 668 lists possible return codes.
Reload function

The Reload function invokes the user exit routine for each segment it reads from the HD Unload data set (DFSUINPT). The user exit routine is invoked one more time at termination. The user exit routine allows you to examine, modify, delete, and control placement of segments before the Reload function places them in the database. The following considerations apply to invoking a user exit routine with the Reload function:

- If the data is compressed in the input file, the Reload function automatically expands the data before passing it to the exit. The Reload function compresses any modified data before placing it in the database.

- Parallel processing is disabled when you specify USEREXIT.

- When you are using the user exit for fixed-length segments with compression and the input to the user exit is compressed or expanded, you should never specify the length field in the work area of the user exit linkage section.

- If the segment data is to be used without change, set the return code to 00. When modifying a segment, copy the segment data from A(segment data) to A(work area), change the data at A(work area), and set the return code to 04.

- The user exit routine must be in a library available to the reorganization reload job step because the Reload function dynamically loads it. The work area for the user exit must be the same size as the largest segment within the DBD.

Table 327 Unload function user exit routine return codes

<table>
<thead>
<tr>
<th>Return code</th>
<th>Routine</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Use the segment pointed to by parameter 2. If no changes are made to the segment before it is placed in the unload file, register 15 must be set to 0.</td>
</tr>
<tr>
<td>4</td>
<td>Use the modified segment pointed to by parameter 3. If the segment is modified before it is placed in the unload file, register 15 must be set to 4.</td>
</tr>
<tr>
<td>8</td>
<td>Delete the segment. When deleting a segment, you must also delete all hierarchically dependent segments; dependent segments should also contain a return code of 8.</td>
</tr>
<tr>
<td>12</td>
<td>Delete the segment and all subsequent segments in this database record. This includes subsequent segments across the hierarchy as well as down the hierarchy. The Unload function deletes the dependent segments under this parent and deletes the rest of this root’s segments that are placed hierarchically after the deleted segment.</td>
</tr>
<tr>
<td>24</td>
<td>Delete the segment and its dependants.</td>
</tr>
<tr>
<td>100+</td>
<td>Terminate the unload process, and pass this return code back as the job step condition code. This can be any three-digit number.</td>
</tr>
</tbody>
</table>
If you are using a user exit to change a key, execute the user exit when the database is unloaded. Specify the NEWDBD and USEREXIT keywords in the Unload function job. Reload the database, and specify HDSORT(YES).

If you are using the user exit to copy storage beyond the end of the segment data, you can specify YES as the third subparameter on the USEREXIT option, as follows:

```
USEREXIT=(exitName,,YES)
```

Specifying YES copies the segment to a 32-KB work area before invoking the user exit routine. A slight CPU usage overhead is associated with copying all segments to another area.

**NOTE**

If you specify NO or a blank for the third subparameter, the segment passed to the user exit remains in its original location. If the user exit tries to address data beyond the end of the segment, you might encounter a system S0C4 abend.

---

**User exit routine register usage**

Figure 103 shows the requirements for register 1 usage on entry to a user exit routine.

**Figure 103  Reload function user exit routine**

```
Register 1 ==> A(segname)
        A(segment data)
        A(work area)
        A(PCB area)
```

**Table 328  Register 1 requirements for Reload function user exit routine (part 1 of 2)**

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A(segname)</td>
<td>Address of a 9-byte area that contains an 8-byte segment name and a 1-byte segment code. At termination, the first eight bytes of the area will contain blanks. Initialization is at the first record.</td>
</tr>
<tr>
<td>A(segment data)</td>
<td>Pointer to the segment data read from the unload data set. The Reload function always expands compressed data before presenting it to the user exit routine. Because the segment is usually physically located in the input data set buffer, the segment length must not be changed in this area.</td>
</tr>
</tbody>
</table>
When you exit the user exit routine, register 15 must contain the decimal return code for the Reload function. Table 329 lists possible return codes.

### Table 329  Reload function user exit routine return codes (part 1 of 2)

<table>
<thead>
<tr>
<th>Return code</th>
<th>Routine</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Register 15 must be set to 0 if no changes are to be made to the segment before placing the segment in the database.</td>
</tr>
<tr>
<td>4</td>
<td>Use the modified segment pointed to by A(work area). If the segment is to be modified before placing the segment in the database, register 15 must be set to 4.</td>
</tr>
<tr>
<td>8</td>
<td>Delete the segment. When deleting a segment, it is your responsibility to also delete all hierarchically dependent segments; dependent segments should also contain a return code of 8.</td>
</tr>
<tr>
<td>12</td>
<td>Delete the segment and all subsequent segments in this database record. This includes subsequent segments across the hierarchy as well as down the hierarchy. The Reload function will delete the dependent segments under this parent and delete the rest of this root’s segments that are hierarchically after the deleted segment.</td>
</tr>
<tr>
<td>16</td>
<td>Put the current HDAM segment into the overflow area.</td>
</tr>
</tbody>
</table>
Reorg function

You can invoke a unload or reload user exit routine during a Reorg function reorganization. The following considerations apply to invoking a user exit routine with the Reorg function:

- Parallel processing for the primary database is disabled when you specify USEREXIT; parallel processing for secondary indexes is not affected.

- If you are using the user exit to copy storage beyond the end of the segment data, you can specify YES as the third subparameter on the USEREXIT option, as follows:

```
USEREXIT=(exitName,U|L,YES)
```

Specifying YES copies the segment to a 32-KB work area before invoking the user exit routine. A slight CPU usage overhead is associated with copying all segments to another area.

**NOTE**

If you specify NO or a blank for the third subparameter, the segment passed to the user exit remains in its original location. If the user exit tries to address data beyond the end of the segment, you might encounter a system S0C4 abend.

Online Reorg function

You can invoke a user exit routine during an Online Reorg function reorganization. The user exit routine is executed during the unload or reload phase and again during the change apply phase. You cannot use an existing user exit for the Unload function or the Reload function; you must modify user exits as described in “User exit routine usage” on page 675. For best performance, BMC recommends that you specify the user exit routine for the unload phase rather than the reload phase.
Restrictions

The following restrictions apply to the Online Reorg function user exits:

- The exit cannot change the key structure or the data within a key.

- The exit must check the PCB area for status codes before doing anything else. If the segment or the change apply log cannot be found, the exit must check the status code.

- The user exit routine function is available in the Online Reorg function only when you do not use an unload BMP.

- The Online Reorg function cannot use existing exits for the Unload function and the Reload function. The Online Reorg function requires different parameters. For more information, see “User exit routine usage” on page 675.

- BMC recommends that you specify SWAP(NO) when you use a user exit with the Online Reorg function. Swapping the data set names manually allows time to refresh applications that access any reformatted data.

Application considerations

When coding an Online Reorg function user exit, you must be aware of the effects of reorganizing a shadow database. The Online Reorg function copies a database and reorganizes the shadow while the original database is online for updates. The Online Reorg function then takes any changes that were made to the original database and applies those changes to the shadow database after the shadow database is reorganized. If you execute a user exit routine during the unload or reload phase, any changes caused by the exit routine are not immediately available for the original database that is available to the online system. The online system still operates on the original database. Any changes made to the original database during the reorganization will be applied to the shadow database during the change apply process, and these changes must be processed by the user exit routine.

The exit must be aware that the change apply process applies changes in a random sequence. Segments may be changed more than once.

WARNING

Invoke user exit routines in the Online Reorg function only after careful consideration. You must exercise extreme caution when using this feature. An appropriate use of a user exit in a Online Reorg function reorganization is to purge segments.
The exit must check which phase it is being executed in (unload, reload, or change apply) so it can take the action appropriate for that phase.

**How the Online Reorg function processes user exits**

During the unload or reload phase, the user exit is executed on the shadow database as it would be during a stand-alone unload or reload. Segments are processed in hierarchical sequence. An invalid return code will result in an abend 4094. At the end of each phase, a termination call is made to the user exit routine. The user exit must close any open DCBs at each termination call.

At the beginning of the change apply phase, the user exit routine is reloaded. The contents of any storage areas will be lost. During the change apply phase, the user exit routine is called once for each time a segment had a REPL, DLET, or ISRT call made to it by the online IMS system. These calls are made in random order. Some segments might be processed by the user exit more than once during the change apply phase. At the end of the change apply phase, another termination call is made to the user exit routine.

If the segment to be replaced in the change apply phase is not found in the shadow database (GE status code), return codes 28 and 32 are the only valid return codes. An invalid return code will result in an abend 4094. For a list of valid return codes, see Table 332 on page 676.

- If the parent of a segment to be re-inserted is not found in the shadow database, the only valid return code is 28. You can determine whether parentage is intact by examining the PCB key feedback area.

- Segments with no keys or non-unique keys cannot be deleted by the exit and then reinserted with a return code 32. The only action the exit can take in this case is to ignore the change that was made to the original database during the reorganization; the only possible return code is 28.

**User exit routines that purge segments**

Some user exit routines purge segments. If a user exit routine that purges segments is executed during the unload or reload phase, the purged segments are not on the reorganized (shadow) database. However, these segments might have been updated on the original database during the reorganization. When the Online Reorg function applies the changes to the shadow database, it cannot locate the segment on the shadow database. The PCB shows a GE status code.
How the Online Reorg function processes user exits

--- EXAMPLE ---

The reload exit purged a segment. Because it was purged, the segment is not on the shadow database. The online system updated the segment in the original database. A change record is generated in the Online Reorg function. The change apply process must update the shadow database using the Online Reorg function change record.

The user exit routine can issue the following return codes:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>Ignore the change from the database that was online and was updated during the reorganization. If the data is needed, the application must save the changes in a user file and process the changes later.</td>
</tr>
</tbody>
</table>
| 32          | Use the segment data that is stored in the A(work area) and put it back into the database. Return code 32 is valid only for unique segments that have keys.  

**Note**: You can put the purged segment back into the database only if the online IMS system made a REPL change and the parent segment is still in the database. If the parent segment was purged, you cannot re-insert any dependent segments.

### User exit routines that reformat segments

Some user exits reformat segments. The segments are reformatted on the shadow database during the unload or reload phase. Because the original database was online and updated during the reorganization, change records will not be reformatted until the user exit routine is executed during the change apply phase. The changed segment’s data is stored in the A(segment change data). The user exit must reformat the segments (the same way segments were reformatted during the unload or reload phase) and place the changed segments in the A(work area). Ensure that the user exit routine issues a return code 4 to tell the Online Reorg function to apply the change from the work area rather than from the change data area.

### Exit for data set name swapping

The Online Reorg function supports a new EXIT that enables you to delay the swap and continue logging until a defined period of time. The EXIT module is DLIOWNDW. No keyword is required to activate this feature. If the module is present, the feature is automatically activated. DLIOWNDW is documented in the sample library.
User exit routine usage

Figure 104 shows the requirements for register 1 usage on entry to a user exit routine. The first five addresses shown in Figure 104 are the same addresses that you use for an unload or a reload user exit routine. The Online Reorg function requires additional addresses, as described in Figure 104.

**Figure 104  Online Reorg function user exit routine**

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A(segment name)</td>
<td>Address of a 12-byte area that contains an 8-byte segment name, a 1-byte segment code, a 1-byte segment level code, and a 2-byte (halfword binary) segment data length. At termination, the segment name will contain blanks and the segment code will contain a binary zero.</td>
</tr>
<tr>
<td>A(segment data)</td>
<td>Address of the segment data from the shadow database.</td>
</tr>
<tr>
<td>A(work area)</td>
<td>Address of the changed segment data to be applied to the database.</td>
</tr>
<tr>
<td>A(PCB area)</td>
<td>Address of the PCB work area that must be checked during the change apply phase.</td>
</tr>
<tr>
<td>A(LPCK)</td>
<td>Address of the logical parent concatenated key A(LPCK) for an unload user exit routine.</td>
</tr>
<tr>
<td>A(phase)</td>
<td>Address of the field that identifies the phase. This field describes when the user exit routine was executed. The following values are valid:</td>
</tr>
<tr>
<td></td>
<td>- UNLD—unload phase</td>
</tr>
<tr>
<td></td>
<td>- LOAD—load phase</td>
</tr>
<tr>
<td></td>
<td>- CHNG—change apply phase</td>
</tr>
</tbody>
</table>

**Table 331  Register 1 requirements for Online Reorg function user exit routine (part 1 of 2)**
When you exit the user exit routine, register 15 must contain the decimal return code. During the unload or reload phase, the return codes are the same as they are for standard unload or reload phase user exit routines, except for return codes of 100+. If you issue a return code of 100+ during the unload or reload phase, the job abends. For possible return codes, see Table 327 on page 668 and Table 329 on page 670.

Table 332 lists possible return codes for the change apply phase.

### Table 332  Online Reorg function user exit routine return codes (part 1 of 2)

<table>
<thead>
<tr>
<th>Return code</th>
<th>Routine</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No action. Proceed with change.</td>
</tr>
<tr>
<td>4</td>
<td>The user exit routine has changed the data passed through the change data area. The changed segment’s data is stored in the work area. Use the work area, rather than the change data area, to proceed with the change apply process.</td>
</tr>
<tr>
<td>8</td>
<td>Delete this segment and all dependent segments. Return code 8 is valid only if the captured call type is REPL and the PCB does not have a GE status code.</td>
</tr>
<tr>
<td>28</td>
<td>Ignore change.</td>
</tr>
</tbody>
</table>
This section discusses COBOL exit routines.

**OS/VS COBOL**

If the user exit routine is written in OS/VS COBOL, you must compile it with the NORESIDENT and NODYNAM keywords. If the OS/VS COBOL program uses the DISPLAY statement, compile it with the ENDJOB keyword. If you use NOENDJOB, each execution of the DISPLAY statement uses additional virtual storage. Eventually, an S80A abend could occur. If the user exit reads or writes an external file, NOENDJOB is required. To prevent a storage problem, limit the number of DISPLAY statements.

**COBOL II**

You might receive positive results by using NORES, NORENT, and NODYNAM as compile options. You might also need to reassemble the macro IGZEOPT with RTEREUS=YES and STAE=NO.

---

**NOTE**

COBOL II exits are called as a subroutine to the Unload function and the Reload function, which are not COBOL II environments. See the IBM COBOL II manuals and any documentation APARs that address this subject.
If your current COBOL II library has IGZEOPT assembled with RTEREUS=NO or STAE=YES, you might want to link your new version of this module to another library. Include this module in the link of your user exit. You might need to include the following COBOL modules in the link-edit of your COBOL II user exit:

- IGZENRI
- ILBOSRT
- ILBOCOM
- ILBOSRV
- IGZEOPT (with RTEREUS=YES and STAE=NO)

For an explanation of the functions that these modules perform, see the IBM COBOL II manuals.

COBOL/370, COBOL for MVS, and COBOL for OS/390

You might receive positive results by using NORENT and NODYNAM as compile options. When using a COBOL/370 exit with the Unload function or the Reload function, you might need to assemble an application-specific copy of module CEEUOPT with RTEREUS(ON) and TRAP=OFF. If your current LE/370 run-time keywords module specifies RTEREUS(OFF) and TRAP=ON, you can assemble a CEEUOPT module with RTEREUS(ON) and TRAP=OFF to another library and INCLUDE it in the link-edit of your user exit. The CEEUOPT module must be assembled with AMODE(31). For details about the LE environment, see the IBM Language Environment manuals.

Using the image copy Data Set Name user exit

When you dynamically allocate an output image copy data set, you can let the Image Copy function determine the data set name, or you can provide a Data Set Name user exit routine to supply the data set name.

NOTE

The Image Copy function passes processing control to the exit routine; therefore, no internal defaults will be used for data set naming conventions. You should provide all data set naming requirements in your routine, especially when you are creating multiple image copies.

No ICPSYSIN keywords are associated with the Data Set Name user exit.
Coding procedures

To use the Data Set Name user exit, provide a load module named ICP$DSN or RCU$DSN.

The Image Copy function searches for the data set name user exit routine in the STEPLIB concatenation. It first searches for a load module named ICP$DSN; if this module is not found, it searches for a load module named RCU$DSN.

If the module is found, the Image Copy function calls the Data Set Name user exit before the dynamic allocation of an output image copy data set. When the exit receives control, registers 0 and 1 point to parameter areas containing information about the data set being allocated. The user exit might examine the information supplied to help it determine the data set name to be used.

The following restrictions apply:

- The exit routine must be reentrant. It may be called asynchronously by multiple image copy tasks.

- The exit routine must save all registers on entry and restore them before returning. On return, register 15 must contain a return code. A return code of zero indicates normal completion of the user exit. A nonzero return code indicates an error; the Image Copy function abnormally terminates the associated image copy task.

- The exit routine must make no assumptions about the addressing mode used at run time. It may be called in either 24- or 31-bit mode and must return in the same mode. The parameter areas passed to the exit routine reside in 24-bit storage.

- The register save area provided in register 13 at entry is prechained. The word at offset 8 from register 13 already contains the address of the next save area available and may be loaded by the exit routine. In any case, the exit routine must not modify the word at offset 8 from register 13 at entry.

$RCUDSN macro

Member $RCUDSN is supplied in the sample library to assist in the creation of the Data Set Name user exit routine. While use of this macro is not required, it is highly recommended because it follows the BMC interface guidelines and restrictions. It provides a powerful and flexible method of generating user data set names.

To create a Data Set Name exit module using the $RCUDSN macro, code an assembly language program using the program skeleton shown in Figure 105 on page 680.
The $RCUDSN macro has one required positional parameter; this parameter can take one of four fixed values—ENTRY, DSECT, EXIT, or DOC—or a variable format. The $RCUDSN statement with fixed values is described below:

- $RCUDSN ENTRY generates standard entry logic for the module, along with standard symbolic register name definitions (such as R0, R1, etc.). $RCUDSN ENTRY is not required; if needed, standard entry logic is generated automatically if $RCUDSN ENTRY is omitted.

- $RCUDSN DSECT generates a DSECT defining the standard parameter area passed using register 0 from the Image Copy function. $RCUDSN DSECT is not required; if needed, a DSECT definition will be generated automatically if $RCUDSN DSECT is omitted.

- $RCUDSN EXIT generates standard exit logic for the module, including appropriate error handling. $RCUDSN EXIT is required; include the $RCUDSN EXIT statement as the standard technique for returning to the Image Copy function.

- $RCUDSN DOC generates a standard block of comments describing the use of the Data Set Name user exit routine. $RCUDSN DOC is not required.

The $RCUDSN statement also has a variable format for the positional parameter which specifies a prototype data set name. This form of the macro may also have up to four keyword parameters as described below. The following illustration shows the $RCUDSN statement with a variable parameter and the corresponding keyword parameters:

```plaintext
$RCUDSN 'prototype',
    FIELDS=fields[,SUB=C'char']
    [,TYPE=type][,STKIC=stkic]
    [,GDG=gdg]
```

**NOTE**

Do not use keyword parameters on $RCUDSN statements with ENTRY, DSECT, EXIT, or DOC as the prototype parameter value.
The SUB, FIELDS, and variable prototype parameters are described below:

- The variable prototype parameter consists of a string of characters which defines a data set name mask. This string must be enclosed in apostrophes. Each character in the data set name mask—except for certain special characters—will be placed into the corresponding position of the generated data set name.

  In the following example, OFFSIT will occupy its current position in the generated data set name:

  $$\text{RCUDSN } '############..####..####..OFFSIT..##'$$

- The SUB keyword parameter defines a single character char which functions as a dynamic substitution character within the mask. A string of consecutive substitution characters defines a single substitution field; the number of substitution characters in the string defines the maximum length of the substitution field. The default substitution character is the pound sign (#).

  In the following example, the pound signs define four substitution fields that will be replaced by variables specified on the FIELDS parameter when the data set name is generated:

  $$\text{RCUDSN } '############..####..####..OFFSIT..##'$$

- The FIELDS keyword parameter defines a list of RX-type addresses. The entire list is enclosed within parentheses, and the items in the list are separated by commas. Each entry in the list identifies the starting address of a character string variable. These variables are used for dynamic insertion of text into the prototype.

  In the following example, the FIELDS keyword parameter will cause the data at symbolic address PREFIX to replace the first set of pound sign substitution characters when the data set name is generated (up to 17 bytes); the data at symbolic address DBD will replace the second set of substitution characters when the data set name is generated (up to 8 bytes):

  $$\text{RCUDSN } '############..####..####..OFFSIT..##'
  \text{FIELDS}=(\text{PREFIX, DBD})$$

**NOTE**

Do not use the vertical broken bar sign (\_) as a substitution character (hexadecimal: X'6A').

In the following example, the pound signs define four substitution fields that will be replaced by variables specified on the FIELDS parameter when the data set name is generated:

$$\text{RCUDSN } '############..####..####..OFFSIT..##'$$

In the following example, the FIELDS keyword parameter will cause the data at symbolic address PREFIX to replace the first set of pound sign substitution characters when the data set name is generated (up to 17 bytes); the data at symbolic address DBD will replace the second set of substitution characters when the data set name is generated (up to 8 bytes):
Dynamic substitution

The prototype data set name mask is scanned from left to right to locate dynamic substitution fields. Each group of consecutive occurrences of the substitution character defines a variable field. The length of the variable field is determined by the number of occurrences of the substitution character. The substitution fields in the prototype string are matched one-to-one with the addresses specified in the FIELDS keyword. The Image Copy function provides several useful fields as shown in Figure 106. Specify any of these fields or any other data available to the exit routine in the FIELDS keyword.

You can use a period as a field separator. Use a single period to separate two adjacent substitution fields, such as in ####.####. To include a period in the generated name between adjacent substitution fields, use two consecutive periods, such as in ####.####. Other period characters not used to separate adjacent substitution fields will be included in the generated data set name as a literal character.

After dynamic substitution has occurred, all leading and embedded blanks are eliminated from the generated string.
Prototype selection

Although you may code multiple statements with the variable prototype format, only one of these $RCUDSN statements is selected during execution to generate the user data set name. The TYPE and GDG macro statement parameters determine which of the multiple statements will be used; the first of the statements that matches the conditions specified by its parameters is the statement selected to generate the user data set name.

The TYPE and GDG macro statement parameters are described below:

- Use the TYPE macro statement parameter to select a prototype based on the type of file being produced. For the Image Copy function, valid types are IC1 through IC10. If you omit the TYPE parameter, the prototype is selected for all output file types.

- Use the GDG macro statement parameter to select a prototype based on whether the output image copy is being generated as a GDG. Specify GDG=Y to select the prototype for generation data sets and GDG=N for ordinary data sets. Omit the GDG macro statement parameter to select both types.

Example

In Figure 107 on page 684 the first prototype format is used if the data set being produced is type IC1 and the data set is not a GDG. The second prototype format is used if the data set being produced is type IC1 and the data set is a GDG.

If none of the prototypes are selected, the exit returns to the main task with a nonzero return code, and the main task indicates that the copy for which the exit was invoked will be disabled. In this example, if two copies are requested, the second copy will be disabled because the exit generates DSNs only for the first copy.

NOTE
For IMS 6.1 and later, the DATE and RTIME fields are the local date and time. If you want to use a GMT date and time, use the GDATE and GRTIME fields. (If you change from use of the DATE and RTIME fields to use of the GDATE and GRTIME fields, you must reassemble and relink the exit.)
In Figure 108 offsets within fields are used for prototype substitution (JOB+3), and the TYPE keyword allows for general comparisons based on the character string provided. For example, with TYPE=IC, the prototype is used for any image copy (IC1 through IC10).

Figure 108  SRCUDSN macro with FIELDS offsets and general TYPE values

In Figure 107 offsets within fields are used for prototype substitution (JOB+3), and the TYPE keyword allows for general comparisons based on the character string provided. For example, with TYPE=IC1, the prototype is used for any image copy (IC1 through IC10).
In Figure 109 assembler coding is used in addition to the $RCUDSN macro. The second prototype includes a user-defined literal string—OFFSIT—in the generated data set name. A field offset reference—TYPE+2—completes the generated data set name.

Figure 109 $RCUDSN macro with Assembler coding

```assembly
RCU$DSN CSECT
$RCUDSN ENTRY
CLC TYPE(3),=C'IC1' IS THIS FOR IC1?
BE ICINAME YES - BRANCH
CLC TYPE(2),=C'IC' IS THIS FOR ANY OTHER IMAGE COPY?
BE OTHER YES - BRANCH
B XITPGM BRANCH TO EXIT - NO PROTOTYPE USED
* THIS WILL CAUSE U4094 ABEND

ICINAME DS OH
$RCUDSN '####################..#####..####..########..######X
##', FIELDS=(PREFIX,JOB+3,TYPE,DBD,DSG)

OTHER DS OH
$RCUDSN '####################..########..########..OFFSIT##'X, FIELDS=(PREFIX,DBD,DSG,TYPE+2)

* THIS CONSTRUCTS A DSN WITH THE FORMAT 'PREFIX.DB.DSG.OFFSIT_', WHERE '_' IS 2-10 (DERIVED FROM THE LAST DIGIT(S) OF 'TYPE')

* RETURN TO IMAGE COPY PLUS
```

---

Appendix E User exits 685
Figure 110 is an example of how to assemble and link the Data Set Name user exit (RCU$DSN or ICP$DSN).

**Figure 110  Assemble and link Data Set Name user exit**

```plaintext
/*+---------------------------------------------------+
/*|                   ASSEMBLE                        |
/*+---------------------------------------------------+
//ASM    EXEC PGM=IEV90,PARM='LOAD,NODECK,BUFSIZE(MAX)'  
//SYSLIB   DD  DSN=SYS1.MODGEN,DISP=SHR,DCB=BLKSIZE=19040
 //    DD  DSN=SYS1.MACLIB,DISP=SHR
 //    DD  DSN=SYS1.AMODGEN,DISP=SHR
//SYSLIN   DD DSN=&&LOADDK,UNIT=SYSDA,SPACE=(CYL,1),DISP=(,PASS)
//SYSPRINT DD SYSOUT=*  
//SYSUT1   DD UNIT=SYSDA,DISP=(,DELETE),SPACE=(CYL,(20,10))
//SYSIN    DD DSN=your.ASM(mbr),DISP=SHR
/*+---------------------------------------------------------+
/*|                       LINK                              |
/*+---------------------------------------------------------+
//LINK   EXEC PGM=IEWL,REGION=0M,COND=(4,LT,ASM), 
 //       PARM='SIZE=(384K,96K),XREF,LREFR,LET,RENT'  
//SYSPRINT DD SYSOUT=*,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSLMOD  DD DSN=bmc.xxx.load,DISP=SHR  
//SYSLIB   DD DSN=your.LOAD,DISP=SHR
//SYSLIN   DD DSN=&&LOADDK,DISP=(SHR,DELETE)
 //    DD DDNAME=SYSIN
//SYSIN    DD *  
MODE AMODE(31),RMODE(ANY)
```
HALDB migration

This appendix defines the level of High Availability Large Database (HALDB) support that BMC offers. The chapter also details the migration process from full-function database to HALDB. This appendix includes the following topics:

Overview ................................................................. 687
BMC HALDB support ................................................. 688
BMC HALDB support limitations ................................. 688
HALDB limitations ..................................................... 688
Migration to HALDB .................................................. 689
HALDB reorganizations .............................................. 693
  JCL ................................................................. 693
  User-controlled options ........................................... 693
  Reports ............................................................. 693

Overview

High Availability Large Databases (HALDBs), introduced with IMS 7, are divided into partitions. A partition is a group of database records that can be treated as a separate database. Each partition has the same capacity as a full-function database. Each HALDB can have a maximum of 1,001 partitions. Each partition can have a maximum of 10 data sets. VSAM and OSAM capacity is 4 GB per data set.

For a detailed explanation of HALDBs, refer to IMS Version 7 High Availability Large Database Guide, published by IBM.
BMC HALDB support

BMC support for HALDBs follows:

- Full-function to HALDB migration process
- Unload function
- Reload function
- Index Build function
- Primary index builds for selected partitions
- Reorg function
- Image Copy function
- MAXM Database Advisor for IMS
- DATABASE INTEGRITY PLUS
- POINTER CHECKER PLUS (hash support)
- Online Reorg function

BMC HALDB support limitations

BMC support for HALDBs has the following limitations:

- Reorganizing HALDBs with secondary indexes requires the Index Build function to properly build indexes.

- Key compression at the root level is not supported.

HALDB limitations

All HALDB limitations apply when using BMC functions with HALDBs. HALDB limitations follow:

- HALDBs are only available with IMS 7 and later. Do not use HALDBs unless you are sure that all sharing IMS systems will remain on IMS 7 or later.

- Non-unique secondary indexes are not supported. Indexes to HALDB must be HALDB indexes (PSINDEX).

- Shared secondary indexes are not supported.

- HALDB partitioning of non-keyed HDAM data is not supported.
Logical relationships between HALDB and full-function databases are not supported.

Symbolic pointers are not allowed.

Batch and utility runs use dynamically allocated information from the RECON.

VSAM data sets must be defined with the REUSE keyword.

## Migration to HALDB

Use the following steps to migrate a full-function database to a HALDB using BMC utilities.

### NOTE

Refer to the IBM publication *IMS Version 7 High Availability Large Database Guide* to determine whether your environment requires additional steps to migrate full-function databases to HALDBs.

1. Required. Image copy the full-function database.

2. Required. Unload the full-function database.

   Use the Unload function to create a BMC LONG format unload data set for a full-function database. Use the standard JCL to create the unload data set used as input to the Reload function; no JCL changes are required.

   If the database has logical relationships that are supported in HALDB, use the FFTOHAL keyword. For more information about the FFTOHAL keyword, see “FFTOHAL” on page 434.
NOTE

If you use the FFTOHAL keyword with the Unload function, you cannot use the REORG command for the Reload function. Dynamic allocation of database files is not supported when the FFTOHAL keyword is used with the Unload function.

The FFTOHAL migration unload might take a long time if the database being unloaded has physically paired bidirectional logical children and at least one side has a very long physical (as opposed to logical) twin chain.

One of the following technique can be used to improve performance:

- Some tuning can be performed by adding a DDCARD to the unload step //$$KEEPVS DD DUMMY and then tuning the //DFSVSAM buffer control cards.

- Add a DD card with the DDNAME matching the paired logical child’s segment name to the unload step causes the utility to defer the creation of the EPS pointer to the load step. The DD card added must point a file approximately large enough to hold all the paired logic twin segments. The DFSURGU0 unload file will also increase in size by a like amount.

  The file created from the DD with the logically paired twin’s segment name must be sorted, loaded into a KSDS, and added to the load step JCL of the logical parent’s database. Note that the KSDS’s are built from one DBD’s unload, but read in the logically related DBD’s load step.

  Sample control cards to be used for the SORT steps and to define the KSDSs will be shown in the unload steps BMCMSG file.

3 Conditional. If you reuse the existing database name or database data set names, BMC recommends that you perform the following steps:

A Use the RECON.BACKUP command to back up your RECON data set.

B Save the original DBD source for the full-function database.

4 Required. Delete all objects related to the old database:

- RECON records (DB, DBDS, and so on)

- Database data sets

- MDA members (for a partitioned database, dynamic allocation only allocates the DBRC-registered data sets)
5 Required. Define HALDB-specific information in the RECON and control block libraries.

A Modify the DBD. The hierarchical structure of the full-function database and the HALDB must be the same. The DBDs must have the same:

- segment names
- parent-child relationships

B Specify HALDB access in DBD. (ACCESS=PHIDAM or PHDAM)

C Remove the DATASET statement. Replace the DATASET statements with the new DSGROUP parameter for all affected SEGM statements.

**NOTE**
Conduct additional DBD modifications specified in the IBM publication *IMS Version 7 High Availability Large Database Guide*.

D Perform DBDGEN and ACBGEN for the HALDB database.

E Use the IBM DBRC Batch Utility (DSPURX00) or the HALDB Partition Definition Utility (PDU) to define partition-specific information for the RECON. When defining the high key values for the associated partitions, the IBM HALDB Migration Aid Utility (DFSMAID0) helps determine how to divide your data across partition boundaries.

6 Required. Define the following HALDB data sets:

- Data set groups associated with the primary database partitions.
- VSAM KSDS for the primary index for each PHIDAM partition.
- VSAM KSDS for the ILDS for each HALDB primary database partition.

7 Required. Load the HALDB partitions.

Use the Reload function to reload the HALDB. All data sets associated with the HALDB partitions are dynamically allocated by using the information stored in the RECON.
The Reload function dynamically determines the format of the input unload data set and converts the data to the required format for loading HALDB partitions. The load process resets the REORG number to 1 and builds WFI records for secondary index relationships that exist in the HALDB. The ILDS data set is initialized empty because all secondary index EPS pointers are automatically healed during the reload.

**NOTE**

The following conditions apply to the Reload function:

- EPS pointers are automatically healed for all subsequent reloads.
- During conversion of full function databases to HALDB, the MONITOR keyword can provide a misleading count. If the migration to HALDB includes empty partitions, the MONITOR keyword reports more records loaded than unloaded because it accounts for the empty partitions.
- The BMC load turns off the partition-INIT-needed flag when reorganizing/migrating all the partitions, so you do not need to run the DFSUPNT0 or DFSUPPRO utility jobs to turn the partition-INIT-needed flag off like you would with some other load utilities.

**8** Required. Rebuild the secondary indexes.

Secondary indexes are rebuilt from the WFI data that the Reload function writes during HALDB partition load. The WFI format is unique to HALDB and can only be used to load a PSINDEX database. Currently, a separate step is required to load the partitioned secondary index data sets after all HALDB partition load functions are complete.

**9** Required. Image copy all partitions.

When HALDBs are defined, the RECON entries for the partition data sets are flagged "defined but not created"; the data sets are only authorized for a Reload function step. After the Reload function step, the data sets are flagged "image-copy required" in the RECON; image copy authorization is allowed but normal processing is not allowed.
HALDB reorganizations

Currently, HALDB reorganizations using BMC utilities are performed similar to full-function reorganizations. The following sections identify the differences.

JCL

Use the same JCL for full-function and HALDB reorganizations. All parameters are dynamically allocated from information in the RECON DD statements. For sample JCL, refer to the chapter that corresponds to the function:

<table>
<thead>
<tr>
<th>Function</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unload function (stand-alone)</td>
<td>Chapter 6, “Unload function”</td>
</tr>
<tr>
<td>Reload function (stand-alone)</td>
<td>Chapter 7, “Reload function”</td>
</tr>
<tr>
<td>Index Build function (stand-alone)</td>
<td>Chapter 8, “Index Build function”</td>
</tr>
<tr>
<td>Reorg function (integrated Unload, Reload, Index Build)</td>
<td>Chapter 2, “Reorg function (Fast Reorg Facility/EP)”</td>
</tr>
<tr>
<td>Online Reorg function</td>
<td>Chapter 3, “Online Reorg function”</td>
</tr>
</tbody>
</table>

**NOTE**
MAXM Database Advisor for IMS supports HALDBs. Use the DBA Toolkit to generate optimized JCL. For more information about MAXM Database Advisor for IMS, refer to the BMC Database Products for IMS User Guide: Advisors and Toolkit.

User-controlled options

The user-controlled options valid for full-function databases are also valid for HALDBs. Note the following parameter requirements for HALDBs.

- DBRC(YES) is required; it is also the DBRC default for HALDBs.
- PDX is not supported.

Reports

Report fields and displays are the same for full-function databases and HALDBs. The only value that appears different is the database type. Database types for HALDBs are PHDAM, PHIDAM, and PSINDEX. Reports are stored in the SYSOUT data set.
Cross-System Image Manager

This chapter describes how to reorganize and copy multiple databases during the same job step. The chapter includes the following topics:

Cross-System Image Manager server address space .................................................. 695
  Prerequisites ............................................................................................................ 696
  Installation considerations ....................................................................................... 696
  Controlling access .................................................................................................. 697
  Default options ...................................................................................................... 697
  Sample JCL procedures .......................................................................................... 701
  Post installation considerations ............................................................................. 702
XIMCHECK .................................................................................................................. 702
  Parameter options .................................................................................................. 703
  Sample JCL ............................................................................................................. 705

Cross-System Image Manager server address space

Cross System Image Manager (XIM) provides the facilities to BMC utility products to exploit the MVS/ESA sysplex environment. These facilities include, but are not limited to, the communication among multiple instances of utility programs and the distribution and execution of utility program processes across all images in a sysplex. By using XIM, these utility products can realize improved workload distribution and reduced elapsed time.

NOTE
XIM is required to reorganize or copy multiple databases simultaneously. For more information, refer to Chapter 5, “Multiple databases.”
Prerequisites

To use XIM you must have the following:

- MVS/ESA Version 4.3 or later
- product LOAD libraries with required APF authorization
- XCF services executing in a multisystem environment
- JES2
- sufficient system linkage indexes (LXs) for your MVS subsystems

Installation considerations

The installation process constructs the XIM started task procedure and the XIM initiator procedure in the HLQ.JCL data set (HLQ is the high-level qualifier that you choose when installing XIM). The default name of the XIM started task procedure is XIMBMC. The default name of the XIM initiator procedure is either the name you entered on the screen as the value for the INIT_PROC option or the default of XIMBMCI. You must copy these two procedures into a procedure library recognized by your JES subsystem. Refer to Figure 113 and Figure 114 on page 702 for examples of XIM procedures.

You must specify the SUFFIX parameter (within the XIM started task procedure) that XIM receives as part of the XIM parameter options member name. The SUFFIX parameter identifies the last one to five characters of a PDS member that begins with the character string XIM (XIMxxxxx). The default name of the SUFFIX is PARMS.

**NOTE**

Note: You do not need to specify a valid SSID parameter within the XIM initiator procedure. XIM generates this value internally.

The installation process constructs a default parameter options member named XIMPARMS in the HLQ.DATA data set. This member contains the default parameter settings for the XIM started task procedure. A PDS data set is required to contain these default parameter settings. Do not use a sequential file. Additionally, all XIM images must reference the same PDS member for startup parameters.
Controlling access

If your site uses RACF or CA-Top Secret security, you can authorize the procedures for the XIM subsystem as started tasks in the started procedures table. If your site uses CA-ACF2 security, you can authorize the procedures for the XIM subsystem as started tasks under the started task control.

Table 333 Authorizing XIM procedures

<table>
<thead>
<tr>
<th>Product</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>RACF or CA-Top Secret</td>
<td>Authorize the procedures for the following subsystems as started tasks in the started procedures table:</td>
</tr>
<tr>
<td></td>
<td>■ XIM performance subsystem</td>
</tr>
<tr>
<td></td>
<td>■ XIM extended job entry subsystem</td>
</tr>
<tr>
<td></td>
<td>If you are running RACF version 2.1 or higher, you can use the STARTED class to add or modify RACF security definitions for started procedures without having to reIPL the system.</td>
</tr>
<tr>
<td></td>
<td>The STARTED class allows you to modify the security definitions dynamically through the RDEFINE, RALTER, and RLIST commands. For more information about using the STARTED class, see the OS/390 Security Server (RACF) Security Administrator's Guide.</td>
</tr>
<tr>
<td>CA-ACF2</td>
<td>Authorize the procedures for the following subsystems as started tasks under the started task control:</td>
</tr>
<tr>
<td></td>
<td>■ XIM performance subsystem</td>
</tr>
<tr>
<td></td>
<td>■ XIM extended job entry subsystem</td>
</tr>
<tr>
<td></td>
<td>For more information, see the appropriate CA-ACF2 publication.</td>
</tr>
</tbody>
</table>

Default options

This section describes how to set the global level or MVS image level parameters for XIM. This section also includes an alphabetical listing of XIM parameters with syntax and valid values.

Global parameters vs. MVS image parameters

XIM provides parameters for establishing the scope of XIM processing in a sysplex environment. You can set all parameters at a global level and some at the MVS image level. When parameters are not set at the global level, XIM default values apply. When values are not set at the MVS image level, the values that are set (or defaulted) at the global level apply.
You can set the following parameters at the global level:

- DUMPLIMIT_TCB
- DUMPLIMIT_SRB
- DUMPLIMIT_API
- INITIATORS
- INIT_PROC
- SYSALLDA
- XCF_GROUP
- XIM_GROUP

Figure 111 shows a sample parameter list that you might provide for XIM. This sample provides values for the global level and provides additional values for two MVS images. The global values apply to all MVS images in the sysplex if you start all XIMs with the same PDS member as input. The MVS image values apply only to the MVS image identified by the system name.

**Default option listing**

**Figure 111  Sample XIM parameter list**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCF_GROUP</td>
<td>XCFBMC</td>
</tr>
<tr>
<td>INIT_PROC</td>
<td>XIMBMCI</td>
</tr>
<tr>
<td>INITIATORS</td>
<td>1</td>
</tr>
<tr>
<td>SYSALLDA</td>
<td>SYSALLDA</td>
</tr>
<tr>
<td>Do SYSA SYSA MVS Image Parameters</td>
<td></td>
</tr>
<tr>
<td>INIT_PROC</td>
<td>XIMIA</td>
</tr>
<tr>
<td>INITIATORS</td>
<td>0</td>
</tr>
<tr>
<td>End</td>
<td></td>
</tr>
<tr>
<td>Do SYSB SYSB MVS Image Parameters</td>
<td></td>
</tr>
<tr>
<td>INITIATORS</td>
<td>4</td>
</tr>
<tr>
<td>End</td>
<td></td>
</tr>
</tbody>
</table>

**DUMPLIMIT_TCB, DUMPLIMIT_SRB, DUMPLIMIT_API**

Global level and MVS image level parameters. These are the three types that XIM uses to categorize an unexpected system abend. Each type is given a limit count of 0 through 100 which may be initialized to the parameter value at XIM start up time. The count is decremented at the time that the particular type of abend occurs. If the count is already zero at the time of the abend, a system dump is bypassed. The intent of these parameters is to prevent multiple dumps from being generated when they are not needed.

This counter can also be modified with the DUMPLIMITS,n,n,n console modify command to change the limits for types TCB, SRB, and API respectively (all three new values must be entered). Also, the DUMPLIMITS command without any following operands will display the current values. For more information, see the *Cross-System Image Manager (XIM) User Guide*. 
The default value for each parameter is 3.

**INITIATORS**

The INITIATORS parameter has the following syntax, where $xxx$ is the number of initiators to start:

```
INITIATORS=xxx
```

Valid values are 0 through 256. The default value is 16.

To prevent XIM initiators from starting on the MVS image, use INITIATORS=0 for that particular MVS image.

The INITIATORS parameter is valid at the global level and the MVS image level. The following definitions apply to the INITIATORS parameter:

- When issued at the global level, the INITIATORS parameter specifies the number of XIM initiators that can be started for each MVS image in the sysplex before the desired work load capacity of the image would be exceeded.

- When issued at the MVS image level, the INITIATORS parameter specifies the number of XIM initiators that can be started for the MVS image before the desired work load capacity of the image would be exceeded.

**INIT_PROC**

The INIT_PROC parameter has the following syntax, where $procName$ is the procedure name:

```
INIT_PROC=procName
```

For the procedure name, use any valid PDS member name.

The default value is XIMBMCI.

The INIT_PROC parameter is valid at the global level and the MVS image level. The INIT_PROC parameter identifies a procedure name that initializes an XIM initiator.

**SYSALLDA**

The SYSALLDA parameter has the following syntax, where $sysallda$ is the MVS system unit name for all DASD devices:

```
SYSALLDA=sysallda
```
This standard name allows you to allocate a data set on any or all DASD devices in your environment. If your systems programmer or DASD administrator has specified a different unit name than the default value of sysallda, you can use the SYSALLDA parameter in your XIM started task procedure to specify the equivalent unit name for your environment.

**XCF_GROUP**

The XCF_GROUP parameter has the following syntax, where `groupName` is the eight-character XCF group name:

```
XCF_GROUP=groupName
```

If you use a group name that is less than eight characters, pad to the right with blanks. Valid characters are A to Z, 0 to 9, and special characters ($, #, and @). To avoid using reserved names, do not begin group names with the letters A through I or the character string SYS. The default value is XCFBMC.

The XCF_GROUP parameter is only valid at the global level. The XCF_GROUP parameter allows multiple XIM subsystems to connect or communicate with each other through the XCF coupling facility or through a CTCA (Channel to Channel Adapter). XIM uses the XCF group name to locate and connect to other instances of itself within the sysplex.

**XIM_GROUP**

The XIM_GROUP parameter has the following syntax, where `groupName` is the eight-character XIM group name:

```
XIM_GROUP=groupName
```

If you use a group name that is less than eight characters, pad to the right with blanks. Valid characters are A to Z, 0 to 9, and special characters ($, #, and @). To avoid using reserved names, do not begin group names with the letters A through I or the character string SYS. The default value is XIMBMCU.

The XIM_GROUP parameter is only valid at the global level. The XIM_GROUP parameter allows the BMC utility products to connect to the XIM subsystem. Currently, you should not change the XIM_GROUP name because it must be identical to the XIM_GROUP name used by the BMC utility product.
**Default options syntax**

Use the following syntax rules when creating or modifying the startup parameter list:

- Use columns 1 through 71.
- Use one parameter per statement.
- Do not continue a parameter onto a second line.
- Do not use duplicate parameters.
- Anything following a parameter and its value is considered a comment.
- An asterisk (*) must precede a comment line.
- The equal sign (=) is the required delimiter.
- Spaces to the left and right of the equal sign are permitted.
- Blank lines (columns 1 through 71) and lines beginning with an asterisk are ignored.

**Default option descriptions**

**Figure 112** shows the sample global parameter list provided for XIM in the HLQ.DATAXIMPARMS data set. This sample provides values for the global level. The global values apply to all MVS images in the sysplex if you start all XIMs with the same PDS member as input.

**Figure 112  Default XIM parameter list**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCF_GROUP</td>
<td>XCFBMC</td>
</tr>
<tr>
<td>XIM_GROUP</td>
<td>XIMBMCU</td>
</tr>
<tr>
<td>INIT_PROC</td>
<td>XIMBMCI</td>
</tr>
<tr>
<td>INITIATORS</td>
<td>16</td>
</tr>
<tr>
<td>END</td>
<td></td>
</tr>
</tbody>
</table>

**Sample JCL procedures**

**Figure 113 and Figure 114 on page 702** show sample JCL procedures.

**Figure 113  XIM procedure (part 1 of 2)**

```bash
//*********************************************************
//*-----------------------------------------------------*
//*        XIM PROCEDURE                                   *
//*-----------------------------------------------------*
//DBUXIM00 PROC
//XIMMAIN  EXEC PGM=XIMMAIN, ACCT=5410,TIME=1439,REGION=0M, PARM='SUFFIX=DBU00'
//* *--- ---------------------------*
//* * STEPLIB MUST BE APF-AUTHORIZED, *
```
Post installation considerations

You should start XIM on each MVS image that will process work for a BMC utility product. To start XIM issue the following command, where ximstc is the name of the XIM started task procedure:

```
s ximstc
```

The default name of the XIM started task procedure is XIMBMC.

XIM provides the facilities to BMC utility products to exploit the MVS/ESA sysplex environment. Each utility product has an options member that specifies certain default SYSPLEX parameters. For descriptions of the SYSPLEX parameters, see the documentation for the specific utility product.

**XIMCHECK**

XIMCHECK is a utility designed to aid in the installation process. It takes a single product distribution library, defined by the SYSLIB DD statement, and searches all LINKLIST data sets and the link pack area (LPA) for the modules contained in the distribution library. In addition, it will search any number of product level data sets (such as distribution data sets residing on the current system or LINKLIST data sets...
or LPA libraries on other systems). For any modules found XIMCHECK will then determine which module is at the highest maintenance level and produce a list of what must be copied where to get the system to the required maintenance level for the distributed product to work.

XIMCHECK, as well as producing the written report, can generate a job stream to accomplish the necessary copies. In addition to producing the list of modules to be copied and the job stream, XIMCHECK can produce an extensive cross reference of the modules that reside in SYSLIB and another location.

### Parameter options

XIMCHECK accepts a number of parameter options to control its behavior. They are explained in this section.

**CHECK/NOCHECK**

The CHECK operand specifies that module regression checking is to be done. The default is NOCHECK. If CHECK is not specified, only the cross reference reports will be generated (the will be no job stream produced, and no regression checking performed).

**XREF/NOXREF**

Specifying XREF will produce a detailed report specifying which level of modules exist in which data sets. The default is NOXREF.

**NOLINKLIST**

The NOLINKLIST parameter specifies that the data sets in LINKLIST and link pack area are not to be Processed. Only those product level data sets represented by the PDS parameter will be processed.

**ALLPDSUPGRADE**

The ALLPDSUPGRADE parameter specifies that all product level data sets are to be upgraded. The default is not to upgrade all of the product level data sets, just the LINKLIST.

**SYSLIB(ddname)**

The SYSLIB parameter specifies an alternate DDNAME for SYSLIB. SYSLIB should represent the distribution data set from the product tape. This must be a single data set only. The default DDNAME is SYSLIB.
Parameter options

**SYSPRINT(**ddname**)***

The SYSPRINT parameter specifies an alternate DDNAME to use for SYSPRINT. The default is SYSPRINT.

**PDS(**ddname**)***

The PDS parameter is used to specify that product library data sets are to be checked, and an optional DDNAME prefix for those data sets. The members of these data sets will be matched with the members from the SYSLIB data sets. There is no default, however if just PDS is specified the default prefix will also be PDS.

**REPORT(**ddname**)***

The REPORT parameter specifies an optional DDNAME to use for the report produced by the XREF option. The default DDNAME is REPORT. Specifying REPORT without XREF is ignored.

**JCLLIB(**ddname**)***

The JCLLIB parameter specifies that a job stream is to be produced. It also allows specification of an optional DDNAME that is to be written to. The default is JCLLIB.

**PREFIX(**prefix1,[prefix2],[...])**

By default XIMCHECK will process all members of the SYSLIB data set. The PREFIX parameter may be used to limit the checking to only those members which match the prefix qualifications. The default is to process all members in the SYSLIB data set.

**RETURN CODES**

Table 334 details the XIMCHECK return codes.

<table>
<thead>
<tr>
<th>Return code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X’00’</td>
<td>No action is necessary because the modules found in existing data sets can be used with the product.</td>
</tr>
<tr>
<td>X’08’</td>
<td>Some modules in to specified SYSLIB should be installed in the LINKLIST. The names of the modules to be included or excluded are printed in the process log.</td>
</tr>
<tr>
<td>X’0C’</td>
<td>All modules in the specified SYSLIB should be installed.</td>
</tr>
<tr>
<td>X’80’</td>
<td>The specified SYSLIB DDNAME was not found.</td>
</tr>
</tbody>
</table>
Sample JCL

The following is sample JCL that may be used to execute XIMCHECK. Sample XIM JCL is also in your sample library.

Figure 115  Sample XIMCHECK JCL

```plaintext
//anyname  JOB (...
//*
//CHECK EXEC PGM=XIMCHECK,PARM='XREF,CHECK,JCLLIB,PDS,NOLINKLIST'
// STEPLIB DD DISP=SHR,DSN=an.xim.product.dataSet
//SYSPRINT DD SYSOUT=*  
//REPORT DD SYSOUT=*  
//JCLLIB DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//SYSLIB DD DISP=SHR,DSN=new.xim.product.dataSet
//PDS001 DD DISP=SHR,DSN=some.xim.product.dataSet
//PDS002 DD DISP=SHR,DSN=another.xim.product.dataSet
```
A

ACB
  Application control block. A control block created from the assembled DBD and PSB during the ACBGEN and PSBGEN processes. It is placed in the ACB library for use during online and certain batch executions of IMS/VS.

ACBGEN
  The process of generating application control blocks.

address space
  The area of virtual storage available for a particular job.

Analyzer
  A component of Online/Defrag that identifies records in the database that meet reorganization criteria.

APF
  Authorized program facility. A facility that identifies programs authorized to use restricted functions.

APF-authorized library
  A library that has been defined in the APF member of the SYS1.PARMLIB.

API
  Application Program Interface. A feature of Unload function and Reload function. In Unload function, the API lets a DL/I application program use the product’s fast database retrieval facilities. In Reload function, the API lets you initially load an IMS database much faster than DL/I can load the database.

authorization profile
  See user authorization profile.

B

bit map
  A coded representation in which each bit or group of bits represents or corresponds to an item.
**block mode**
An online reorganization method in which every record in the specified database is reorganized.

**block free space factor**
A reorganization parameter that specifies the minimum percentage of free space within a block to qualify it as a reorganization target.

**block improvement percentage factor**
A reorganization parameter that specifies the anticipated percentage of improvement in used blocks that must be met to qualify a record for reorganization.

**BMP**
Batch message processing. A mode of program execution within an online IMS environment. It is accomplished by a batch message processing program that has access to online databases and message queues.

**buffer**
A routine or storage area used to compensate for a difference in rate of data flow, or time of occurrence of events, when transferring data from one device to another.

**C**

**CI**
Control interval. The unit of transfer used by VSAM between main storage and direct access devices. Logical records are grouped and recorded by VSAM within control intervals.

**command**
An order for an action to take place or a statement used to request a function. A command consists of the command name and its parameters.

**compression**
A process that eliminates free space, empty fields, redundancies, and unnecessary data to shorten the length of records or blocks in a data set.

**control statement**
A statement in a job step that describes an action or task to be performed.

**CSECT**
Control section. The part of a program specified to be a relocatable unit, all elements of which are to be loaded into adjoining main storage locations.

**D**

**DASD**
Direct access storage device. A device that contains data and allows random read and write access.
data set group
A subset of a database with one or more unique segment types.

data set name swapping
The process of swapping data set names after a reorganization is complete. Online Reorg function and Reorg function can swap the data set names automatically, or you can swap them manually.

database
A collection of data with a given structure for accepting, storing, and providing data for multiple users. IMS databases are organized in a hierarchical manner to eliminate data redundancy.

database record
A root and all of its dependents.

database recovery
The process of restoring a corrupted physical database data set to a point in time before the corruption occurred.

Database Selection facility
The Database Selection facility (DBSF), available in Unload function, allows you to retrieve selected database records from a database and print them or write them to an HD Unload data set.

DBCTL
Database control. The IMS database control subsystem.

DBD
Database description. A description of the physical and logical characteristics of a DL/I database. It describes attributes such as the database organization and access method, the segments and fields in a database record, and the relationship between types of segments. The DBD is created by coding a series of DL/I macros which are then assembled and link-edited into a DBD library. DBDs can be used during batch processing of databases and as input to the ACBGEN process.

DBRC
Data Base Recovery Control. A feature of IMS that maintains information required for database recoveries and logging control.

DBSELECT
The command that invokes the Database Selection Facility (DBSF).

DBSF
Database Selection facility.
**descriptor code**
A character string that indicates the means of message presentation and deletion.

**dependent segment**
A segment in a hierarchical database that relies on a segment at a higher level for its full definition.

**DFSRESLB**
The library of IMS modules.

**DFSMDA library**
The IMS library containing dynamic allocation members.

**DL/I**
Data language/1. A data manipulation language that provides an interface between user applications and IMS.

**DL/I database**
A database that is created and accessed using DL/I and uses one of the following database organization methods: HSAM, HISAM, SHISAM, HDAM, HIDAM, SHSAM, GSAM, MSDB, or DEDB.

**DMB**
Data management block. An IMS control block in main storage that describes and controls a physical database. It is constructed from information obtained from the ACB library or the DBD library. DMBs reside in main storage and can be used during IMS online processing.

**DSECT**
Dummy control section. A control section that an assembler can use to format an area of storage without producing any object code.

**dynamic allocation**
A feature that allows a program to allocate data sets based on the resources currently needed and information from control blocks, rather than from DD statements.

**E**

**EPS**
Expanded prefix segment. Contains information regarding indirect pointers. Logical child segments and secondary index segments require an EPS.

**ESDS**
Enter-sequence data set. A VSAM data set in which records are loaded with respect to their contents. ESDS relative byte addresses cannot change.
EXCP
   Execute channel program. An operating system macro instruction used by the physical IOCS to start execution of a single channel program. It normally refers to the number of I/O operations initiated to perform an action, such as reading a work file or writing a file to disk or tape.

exit processing
   The series of system activities that occur when disconnecting from common services.

Extractor
   The component of Online/Defrag that extracts information from the primary database and prepares database data sets for analysis by the Online/Defrag Analyzer.

F

fragmentation
   A condition in which record storage is scattered among numerous blocks. The disorganization increases I/O activity and reduces operating efficiency.

free space determination
   The process by which a block is evaluated to determine if it meets the minimum requirements to qualify it as a target block. The free space determination parameter is defined by the FSPF keyword.

$GLBL
   The name of the global options module for the Reorg function.

full-function database
   A database that is created and accessed using DL/I and uses one of the following database organization methods: HSAM, HISAM, SHSAM, SHISAM, HDAM, or HIDAM.

G

GSG
   Global service group. A feature in IMS Version 5.1 that enables you to specify groups for remote site recovery.

H

HALDB
   High Availability Large Database. A partitioned full-function database. Maximum number of partitions: 1001. Database types include: PHDAM, PHIDAM, and PSINDEX.

HD
   Hierarchical direct. A form of physical storage organization in which database segments that represent a physical database record are related by direct address pointers in the segment’s prefix.
HDAM
Hierarchical direct access method. A type of database that consists of a single VSAM ESDS or OSAM data set. Segments are referenced by their relative byte address within the data set. Root segments are referenced through the use of a randomizing routine.

HD Sort utility
A utility included with Reload function that enables you to sort data before reloading it.

HF
Hierarchical forward pointer.

HIDAM
Hierarchical indexed direct access method. A type of database that consists of a VSAM ESDS containing the principal database and a VSAM KSDS containing its associated index. Segments are referenced by their relative byte address within the ESDS. Root segments are referenced through the KSDS index.

hierarchic sequence
In IMS databases, the sequence of segment occurrences in a database record defined by traversing the tree, from top to bottom, front to back, and left to right.

hierarchical database
A database that is organized in the form of a tree structure in which each record or sector has only one owner. The structure represents how the records or segments are interrelated, and it predetermines the access paths to data stored in the database.

hierarchical scan
A traditional database scan, where the segments are scanned hierarchically for index records.

HISAM
Hierarchical indexed sequential access method. A type of database that consists of a VSAM ESDS and a VSAM KSDS. Segments are referenced either sequentially or directly through the KSDS data set. The ESDS is used to contain overflow segments.

HSAM
Hierarchic sequential access method. A database access method for sequential storage and access of segments on tape or direct access storage.

ICNEEDED flag
The DBRC Image Copy Needed flag.

IDCAMS
The VSAM access method service. A facility used to define, delete, and reproduce VSAM data sets.
ILDS
Indirect list data set. A repository for indirect pointers used for PHDAM and PHIDAM databases. Each HALDB partition has one ILDS.

image copy
The process of creating a duplicate or backup copy of a database data set.

image copy data set
A data set created using the image copy process.

IMS/ESA

IMS/VS
Information Management System/Virtual Storage. A database/data communication system that manages hierarchical databases and networks.

IMSID
IMS identification. A unique, 4-letter identifier for a specific IMS system and its associated regions.

indexed database
See primary database.

input data sets
The disorganized data sets used as input to the database reorganization.

I/O
Input/output operation. The transfer or retrieval of data between processor storage and peripheral equipment.

IPL
Initial program load. The procedure that causes an operating system to initialize for operations.

J
JCL
Job control language. A control language used to identify a job to the operating system and describe the job’s requirements.

K
key
A unique identifier associated with a database record that indicates the placement of the record in an indexed file.
key list data set
A data set that contains a table of record keys. The key list data set identifies records that are recommended for reorganization.

keyword
A name or symbol that identifies a parameter.

keyword parameter
A parameter that is identified and understood by the system by a keyword and one or more values. Keyword parameters do not have to be coded in a specified order.

KSDS
Key-sequenced data set. A VSAM data set in which records are loaded in key sequence and controlled by an index.

LDPPARMS
The name of the global options module for Reload function.

LIBDEF
A feature of ISPF that provides for the dynamic definition of application data sets, thus allowing application data sets to be specified during an ISPF session. This eliminates the need for allocation statements to define all application data sets prior to invoking an ISPF session.

LNKLST
A member in an MVS system data set that defines libraries that should be accessed during job processing.

logical relationship
A user-defined path between two independent segments. Logical relationships can be in the same database or different databases.

MDALIB
The library of dynamic allocation members.

minimum block improvement
A reorganization parameter that specifies the minimum number of blocks that are expected to be reclaimed to qualify a record for reorganization.

MVS
Multiple virtual storage. The control supervisor under which IMS executes. Synonymous with MVS/XA, MVS/ESA, and MVS/370.
Online/Defrag Job
A component of MAXM Reorg Solutions for IMS that processes the database to perform a reorganization.

online reorganization
A database reorganization that features concurrent updates and no downtime.

option
See keyword.

OS/390
An integrated network computing-ready operational environment consisting of more than 70 base elements and integrated optional features delivered in an integration-tested system. Synonymous with MVS.

OSAM
Overflow sequential access method. A database access method that combines selected characteristics of BSAM and BDAM for handling data overflow.

parameter
A variable used with a command to affect its result.

partition
A fixed-size division of storage.

partitioned database
A database that is divided into partitions which can be treated as separate databases. Partitioned databases include the IBM product IMS/ESA Partition DB (divisible into 32 partitions) and HALDBs (divisible into 1001 partitions).

PCB
Program communication block.

PCF
Physical child first pointer. A segment prefix pointer in a physical parent used by DL/I to access the first occurrence of a dependent physical child segment.

PCL
Physical child last pointer. A segment prefix pointer in a physical parent used by DL/I to access the last occurrence of a dependent physical child segment.
PDS
Partitioned data set. A data set in direct access storage that is divided into portions, called members, each of which can contain a program, part of a program, or data. PDS is synonymous with library.

PDX data set
A BDAM preformatted data set that contains entries in a directory and their associated data records. The PDX is unique to BMC products.

PFX
Prefix. A high-level qualifier.

PHDAM
Partitioned HDAM database, see HDAM.

PHIDAM
Partitioned PHIDAM database, see HIDAM.

pointer
A 4-byte value that contains the address of a segment.

post processing
The phase in Online Reorg function and Reorg function where data set names are swapped, DBRC commands are executed, and the change logs are recovered. If automatic data set name swapping was specified, this phase is invoked automatically. You can start (and restart) this phase by executing the RESTART command.

PP
Physical parent pointer. A segment prefix pointer used by DL/I to relate a physical child segment to its physical parent.

primary database
The database from which index records are extracted. The index records make up the secondary index or primary index of a HIDAM database. The primary database is also known as the target or indexed database.

PSB
Program specification block. A PSB is a control block that relates a DL/I application program with the databases which it accesses. A PSB consists of one or more PCBs and is built during the PSBGEN process.

PSBGEN
The process of building a PSB by coding a series of DL/I macros, which are then assembled and link-edited into a library.
PTB
Physical twin backward pointer. A segment prefix pointer in a segment, used by DL/I to access the previous occurrence of the same type of segment. Each segment belongs to the same physical parent.

PTF
Physical twin forward pointer. A segment prefix pointer in a segment, used by DL/I to access the next occurrence of the same type of segment. Each segment belongs to the same physical parent.

R
RAA
Root addressable area. The portion of an HDAM database that contains root anchor points from which root segments can be chained. The number of CIs or blocks in the RAA is specified within the DBD.

randomizer
A program that is used to assign a root segment to a specific root anchor point in the RAA of an HDAM database.

RAP
Root anchor point. A pointer maintained by DL/I within HDAM database CIs or blocks. Each RAP can contain a pointer to a root segment or chain of root segments.

RBA
Relative byte address. The hexadecimal value of the record’s relative byte address.

RBN
Relative block number. A number that indicates the location of a block within the data set.

Recommendation Selector
Online/Defrag component that analyzes a database and recommends the reorganization parameters for an Online/Defrag reorganization that will result in the greatest decrease of I/O events (buffer searches).

RECON data set
The data sets DBRC uses to contain information about logging activity and events (such as image copies) that can affect the recovery of databases. See also DBRC.

record
See database record.

Record Analysis
An Online/Defrag process that collects information about a record to determine the level of record fragmentation. The process involves the extraction and analysis of database records. See also Extractor and Analyzer.
Reorg function reorganization
An Reorg function reorganization in which you specify INPUT(DB) or accept it as the default.

reorganization target
A block that is available to receive reorganized record segments.

RESLIB
The IMS library data set that contains the authorized load modules for the IMS system.

Restart facility
A utility available in MAXM Reorg/Online and Reorg function that allows you to restart post-processing tasks.

root anchor point
A pointer maintained by DL/I within HDAM database CIs or blocks. Each RAP can contain a pointer to a root segment or chain of root segments.

root key
The key of the root segment.

root segment
The highest level segment in the tree structure of a hierarchical database.

routing code
A code used to direct messages to a defined output location.

RSR
Remote site recovery. A set of features in IMS Version 5.1 and later that provide a way to restore IMS databases and services at a remote site when the active site becomes unavailable.

scan mode
An online reorganization method in which the primary database is scanned and every record that meets reorganization criteria is reorganized.

secondary index
A DL/I database that allows additional access paths to a subject database record. Secondary indexes enable direct or sequential processing of database records on fields that are not root segment key values.

segment
The smallest unit of access to an IMS/VS database. Segments can be root segments or dependent segments.

segment code
A code that defines to IMS a segment’s position in the database hierarchy.
sequential scan
A feature of Index Build function that scans database blocks sequentially, rather than hierarchically, to gather index records. A sequential scan can complete significantly faster than a hierarchical scan.

shadow data set
A temporary database data set that is identical to the permanent database data set. The shadow data set accepts and logs updates during the reorganization of the permanent database. After the reorganization, the permanent database processes the updates log from the shadow data set.

shared secondary index
A single index database that contains up to 16 secondary indexes.

SHISAM
Simple hierarchical indexed sequential access method. A HISAM database containing only one segment type.

SIUPARMS
The name of the global options module for Index Build function.

SMS storage
The IBM Storage Management Subsystem.

STEPLIB
The execution library of concatenated programs.

symbolic pointer
The order of segment keys that must be retrieved to reach a particular segment.

T

target block
A block that meets the minimum requirement of free space to receive a reorganized record. Target blocks are identified during the free space determination that occurs throughout the online reorganization.

target database
See primary database

TSO
Time sharing option. An operating system option that provides interactive time sharing from remote terminals.

U

ULPPARMS
The name of the global options module for Unload function.
user authorization profile

A method to limit access to and use of specific BMC product functions and components.

user exit

A feature that allows execution of user-written code at specific points during the execution of a process.

user ID

A string of characters that uniquely identifies a user to a system.

V

VOLSER

Volume serial.

VSAM

Virtual Storage Access Method. An access method for direct or sequential processing of fixed and variable length records on direct access devices.

W

WTO

Write-to-operator. An optional feature that allows a product to write a message to the system console operator to inform the operator of errors and unusual system conditions that could require correction.

WTOR

Write-to-operator with reply. An optional feature that allows a product to write a message to the system console operator to inform the operator of errors and unusual system conditions that could require correction. The operator can respond with requested input data.
Index

Symbols

#LDPGLBL 222
#SIUGLBLx 253
#ULPGLBL 184
$BMCSMOD DD statement 620
$RCUDSN macro 679

Numerics

8-GB database 500

A

abbreviated records 427
abend codes 617
See also interrupting and resuming an online reorganization
ABNDNO keyword 406
ACBLIB 130
ACBSYSIN 130
access method options for image copy data sets 331
accumulated database record bytes keyword 528
add segments during reorganization 494, 497
address space, sort and load indexes 542
AGN keyword 407
AIB interface 200
AIC command 312, 313, 349, 389
allocation of image copy data sets 325
ALLOWLC keyword 407
ALLOWLP keyword 407
AMP with VSAM KSDS buffers 186
AMSOUT DD statement 351, 408
AMSOUT file 408, 631
AMSOUT global option 408
AMSPDS DD statement 352
analysis report, See recommendation report
Analyzer 144, 150
ANLPEKEYO DD statement 151
ANLPRINT DD statement 151
AOI
  Online Reorg function 126
  Reorg function 409
AOI keyword 409
AOIUSER keyword 410

APITRACE DD statement
diagnostics 622
 Reload function API 237
 Unload function API 203
application group name keyword 407
Application Interface Block (AIB) 200
application processing, DBRC authorization 640
application program interface. See Reload function API
and Unload function API
APPLICATION RESTART CONTROL 114
AR/CTL 114
AUTHERR keyword 411, 412
authorization
  action Reorg function takes if DBRC prohibits authorization 495
  authorization error option 411, 412
  DBRC 104, 640
  failures 411, 412
  failures for DBRC 618
automated operator interface. See AOI
automatic data set name swapping 549, 551

B

BACKOUT.NEED flag 641, 645
Bad Block/Pointer report 585
banner page, reports 160
batch region controller
  Reload function 212
  Unload function 174
BEGIN keyword, Database Selection facility 193
bidirected technique of HDAM space search 533
block mode 139, 159
block size for image copy data sets 333
block truncation example 557
BMC Software, contacting 2
BMC.MXE.LOAD library, placement in STEPLIB
concatenation
  for Reload function API 236
  for Unload function API 206
BMC.MXE.SAMP library, PRPPARMS member 283
BMCFLAGS, USERHDR subkeyword 571
BMCMSG DD statement 147, 151, 155, 190, 353
BMCPRINT DD statement 151, 155, 191, 354
BMCPRINT keyword 522
BMCTRACE DD statement 151, 156, 355, 621
BMPs 114
buffer space keyword 414
buffers
   DFSURGU1 186
   image copy data sets 332
   Unload function 186
   VSAM KSDS 186
BUFNI/BUFND option 186
BUFSIZE keyword 447
BUFFP option 186
BUFSIZE keyword 414, 577
BUILD command 389
   See also Index Build function
build indexes by scanning primary database 257
build indexes in parallel mode 504
build primary index 258, 520
build secondary indexes 534
BUILDALL command 389
bypass PDX usage 371
bypass retrieval of LPCKs 487

C
cataloged image copy data sets 330
CATDS parameter in DBRC 330
chain buffer length keyword 447
change DBD definition 496
CHANGE RECORDING FACILITY return codes 618
changes to product 32
check physical pointers 524
check RECON data set keyword 414
CICS/DBCTL environment. See the BMC Products for IMS
   Installation Guide
CKUPDATE keyword 414, 577
COBOL II user exit 677
COBOL/370 user exit 678
codes
   abend 617
   completion 617
   descriptor 491
   return 617
   routing 491
commands
   AIC 312, 313, 349, 389
   BUILD 389
   BUILDALL 389
   DBSCAN 191
   DBSELECT 193
   GBL 312, 313
   GROUP 312, 313, 335
   GROUP subcommand 168
   IC 312, 313, 349, 389
   ICPSYSIN 312
   INIT.RECON 330
   MVS MODIFY 491
   NOTIFY.IC 331
   OREORG 141
   overview 388, 389
   RELOAD 389
   REORG 389
   SORT, HD Sort utility 232
   syntax 405
   TRACE, Unload function API 205
   UNLOAD 206, 389
   USERLOAD, Reload function API 237
   USERREAD, Unload function API 204
   commands data set 372
   comments in PLUSIN data set 405
   completion codes 617
   COMPRESS keyword 225, 415, 419, 576
   compressed image copies 76
   compression
      compressed data and user exit routine 665
      compressed root keys not supported by LIMITS
         keyword 483
      compressed segments 416, 419
      compressed segments in HD Unload file 417
      compression routines and LIMITS keyword 483
      formula for compression factor 616
      restrictions for compressed keys 74
      when HD Unload data set is created by Unload
         function 417
      when HD Unload data set is not created by Unload
         function 417
   Concurrent Prefix Update
      JCL example 278
      overview 277
   Concurrent Reorganization report 586
   conditional reorganization feature in Advisor 419
   conditional reorganizations 199
   console interface, JCL generator 38
   continuation characters 406
   continue after sequence error 537
   control statement data set
      database scan 191
      Database Selection facility 193
      DD description 372
      HD Sort utility 232
      Index Build function 248
      Online Reorg function 123
      Reload function 215
      Reload function API 237
      Reorg function 93
      Unload function 177
      Unload function API 203
   control statement syntax 404
   conventions, documentation 32
   copying multiple databases 165
   COUNT keyword, Database Selection facility 193
   create ddnames in pairs 562
   create empty database 103, 226
create empty secondary index 262
create new secondary index 260
creating an HSAM database 448
CREORG keyword 419
CRFRDER DD statement
data set name 483
description 356
SMS 485
space to use for dynamic allocation 486
unit name 486
with Online Reorg 117
CRFRDER2 DD statement 117, 356
CRFUXASM user exit routine example 664
CRFUXCOB user exit routine example 664
Cross-System Image Manager 167, 695
customer support 3
customized user output file 568

D

DASD use for image copy data sets 328
DATA PACKER/IMS 483
Data Set Buffer Statistics report 587
Data Set Group Summary report 588
data set groups, Reorg function JCL considerations 96
data set name swapping
and logical relationships 629
description 51
swap data set names manually 630
SWAP keyword 549
using input ddnames in pairs 562
using substitution characters 560
Data Set Name user exit 330, 664, 678
Data Set Summary report 605
data sets, PDX 143, 156, 529
database recovery 133
database reorganization 135
database restructure 494
database scan 187, 189
Database Selection facility
command examples 195
description 192
output in printed format 197
Database Summary report 586
Database Utilities Subsystem, use with NOTIFY option 496
database-specific options member
creating 324
use of 319
Dataset List report 591
Dataset Statistics report 591
Datasets Used report 592
DBA Toolkit
automated data collection 61
automatic analysis 62
automatic configuration 62
conditional reorganization 62
control block disassembly 62
Database Management Console 64
DBD mapping 63
exception management 63
features and benefits 61
JCL generation 63
repository management 63
statistical information 64
DBA Tools
DBCTL environment. See the BMC Products for IMS 
Installation Guide
DBD Change Summary report 592
DBD changes
Online Reorg function 129, 496, 498
Reorg function 496, 497
DBD keyword
description 420
HD Sort utility 232
not available with REORG command 576
DBD names, specify DBD name of primary database 420
DBD Summary Report 162
DBD Summary report 592
DBDLIB 370
DBDNAME, USERHDR subkeyword 571
DBDSGRP keyword 337, 421
DBIL-type reorganization reports 608
DBR keyword 421
DBRC
abend code for authorization failures 618
abend code for signon failures 618
action Reorg function takes if DBRC prohibits
authorization 495
and High-performance Prefix Resolution function 274
and Index Build function 642
and Reload function 213
and Reorg function 643
and the Online Reorg function 39
and Unload function 175
authorization 104, 640
authorization failures 105, 411, 412
authorization for application processing 640
CATDS and NOCATDS parameters 330
interface with PRPURG0 297
invoke DBRC 422
NONRECOV databases 643
notifications 103, 125, 644
requirements 71
role in reorganization 637
DBRC keyword
description 422
not available with REORG command 576
DBRC keyword (Prefix Resolution/Update function) 423
DBR-type reorganization reports 609
DBSCAN command 191
DBSELECT command
   Database Selection facility 193
   examples 195
DBSF. See Database Selection facility
DBUDSEH1 module, used by Reload function 301
DBUSS, with USE NOTIFY option 496
DBVTOTOC 76
DCB information for output data set 285
DD statements
   AMSOUT 351, 408
   AMSPDS 352
   ANLPKEYO 151
   ANLPRINT 151
   APITRACE 622
   BMCMSSG 147, 151, 155, 353
   BMCPRTINT 151, 155, 354
   BMCTRACE 151, 156, 355
   CRRFRDER 356
   CRRFRDER2 356
   DFSRESLB 147, 150, 155, 357
   DFSUINnn 358
   DFSUINPT 358
   DFSURCD 359
   DFSURGU1 360
   DFSURGU2 360
   DFSURGUn 361, 362
   DFSURIDX 363
   DFSURWF1 364
   EXEC 150, 154
   HLDURIDX 367, 368
   ICPSYSIN 93, 118, 310
   IEFFRDER 369
   IEFRRDER2 369
   Image Copy function 310
   IMS 147, 151, 155, 370
   OLRKEYS 155
   output image copy data set 325
   PCPDANIP 151
   PCPDO001 147
   PCPDO001 147
   PCPSYSIN 148, 334
   PDX 156, 371
   PLUSDMP 311, 620
   PLUSIN 148, 152, 155, 372
   PLUSLIST 373, 620
   PLUSOUT 311, 373, 621
   print data set 374
   PFRURIDX 375
   PFRURWF1 375
   RECONx 155, 376
   REPORTS 311
   SEQERROR 377
   SORTIN 377
   SORTWK 378
   STEPLIB 146, 150, 155, 378
   SYSOUT 379

   SYSPPRT 353
   SYSUDUMP 147, 151, 380
   xxxPRINT 374
   DDN keyword 423
ddbname prefix for second sort keyword 447
ddnames
   create in pairs 562
   substitution characters 560
delete segments during reorganization 494, 497
DELTA IMS DB/DC 130
DESC keyword 424, 490
DESCCODE keyword
   description 491
   HD Sort utility 232
descriptor codes 491
DFFORTS 311
   DFFORTS member of PDX 181, 219, 280
DFSDBUX1 keyword
   and HD SORT 445
   description 425
   HD Sort utility 232
dfSfPARm DD statement, HD Sort utility 231
DFSRESLB DD statement
   Analyzer 150
description 357
   Extractor and Recommendation Selector 147
   online reorganization 155
   Reload function API 236
   Unload function API 203
DFSRRCC0 75, 174, 187, 201, 212, 235
DFSUINnn DD statement 358
DFSUINPT DD statement 97, 358
DFSURCDs DD statement 190, 359
DFSURG10 (High-performance Prefix Resolution function)
   274
DFSURGL0 75, 212
DFSURGP0 75
DFSURGSO module, used by IMS HD Scan 301
DFSURGU1 buffers 186
DFSURGU1 DD statement 97, 360
DFSURGU1 keyword
   description 425
defect on performance 187
   not available with REORG command 576
DFSURGU2 DD statement 97, 360
DFSURGU1 DD statement 361, 362
DFSURIDX DD statement 363
DFSURWF1
   load indexes from existing DFSURWF1 file 257
   parameter for INDD 472
   Reorg function 96
DFSURWF1 DD statement
   database scan 191
   Reload function 364
DFSVISAMP DD statement
   Online Reorg function 118
   Unload function API 203
diagnostics
  $BMC$MOD DD statement 620
  APITRACE DD statement 622
  BMCTRACE DD statement 621
  diagnostics data set 355
  return codes 617
  SYSUDUMP DD statement 621
Disassembled DBD report 594
Distribution of Database Record Sizes report 526, 594
Distribution of Free Space report 595
Distribution of HDAM RAP Chain Lengths report 526, 595
DL/I conditions under which Unload function API works 207
DL/I status codes
  Reload function API 239
  Unload function API 207
DLIGSET0 647
  See also the BMC Products for IMS Installation Guide
DLIMPARM 140, 143
dual HD Unload data sets and LIMITS keyword 483
dual logging 356, 369
DYNALLOC keyword 430
dynamic allocation
  database data sets 430
  image copy data sets 326
  Image Copy function 326
  logs 483
  reports data set 522
  SMS image copy 329
  tape image copy 327

E

electronic documentation 31
empty database, create 103, 226
empty secondary index, create 262
END keyword, Database Selection facility 194
End of Job Access keyword 433
environment setup member 647
  See also the BMC Products for IMS Installation Guide
EOJACCES keyword 433
error
  continue after pointer error 524
  continue after sequence error 537
examples
  Database Selection facility 195
  HD Sort utility 233
  Index Build function 248
  Online Reorg function 123
  Reload function 215
  Reorg function 93
  Unload function 177
  Unload function API
    TRACE command 206
    USERREAD command 205
exclusive access 540
EXCP access method 332
EXEC DD statement 150, 154
EXEC statement
  database scan 190
  description 347
  HD Sort utility 230
  Reload function API 236
exit. See user exit routine
expand compressed segments option 415, 419
EXPAND keyword 187, 415, 417, 419, 577
expiration date for logs 485
EXTENDED BUFFER MANAGER for IMS 204, 205, 574
extracting data
  Extractor 144
  keywords 148
  Extractor 144
  extra-short records 427
  F

FBFF keyword 577
features and benefits
  Index Build function 55
  Online Reorg function 48
  Online/Defrag function 60
  Prefix Resolution/Update function 58
  Reload function 54
  Reorg function 49
  Unload function 52
FEOV keyword setting 421
FFTOHAL keyword 434
field-level sensitivity, Reload function API restrictions 240
File Size keyword 437
files not needed with Reorg function 97
FILLERnn, USERHDR subkeyword 571
FILSZ keyword 437
FIXED keyword 285, 437
fixed-length segments and user exit routine 665
fixed-length sorting 575
flags
  BACKOUT.NEEDED 641, 645
  ICNEEDED 457, 641, 642, 644, 645
  NOAUTH 104, 641, 645
  RECOV.NEEDED 641, 645
flip-flop technique of HDAM space search 533
FMTRAA keyword 225, 438, 439, 577
force DBRC keyword 423
force end-of-volume 421
force logical parent counter field keyword 439
FORCELP keyword 285, 439
format of HD Unload records 425
format the root addressable area 438, 439
free space analysis information 335
free space determination 136
free space factor keyword 441
free space percentage factor (FSPF) 443
FREQ keyword 335, 440
FRF$GLBL module 385
FRF/EP
   JCL requirements (for logically related databases) 88, 89, 170
   logical relationships 629
   return codes 618
FRFPRINT DD statement 374
FRFPRINT keyword 522
FRFREST0 631
FRMT1 records 428
FRMT2 records 429
FRMT3 records 429
FSPF 443
FSPF keyword 441, 577
Full Function to HALDB keyword 434
FUNC keyword 337

H

HALDB
   JCL 693
   keywords 693
   migration 689
   PART keyword 508
   primary indexes 259, 470, 521
   reorganizations 693
   reports 693
   support 688
   support limitations 688
   hardware requirements 70
   HD Scan 301
   HD Sort - Record Counts report 599
   HD Sort utility
      description 227
      example 233
      HDPRINT keyword 522
      invoking with HDSORT keyword 444
      JCL requirements 229
      keywords 232
      overview 227
      required for certain changes 74
      system flow 229
   HD Unload file as input to Reorg function 84
   HD Unload file used as input to Reorg function 474, 475
   HD Unload record format 425
   HDAM databases
      and USEINDEX keyword 565
      change database randomizer 224
      convert to HIDAM 53, 565
      Database Selection facility BEGIN option 193
      Database Selection facility END option 194
      DBD restrictions 74
      empty database 226
      insert rules for IMS 224
      insert rules for Reload function API 240
      LIMITS keyword 481
      Reload function API restrictions 240
      Reorg function JCL considerations 95
      restrictions 74
      space search algorithms 532
      space search method
         bidirected technique 533
         flip-flop technique 533
   HDAM overflow area 503
   HDAM randomizer 74, 224, 497
   HDAM Root Addressable Area Statistics report 597
   HDAM Root Placement Relative to Its Home Block report
      526, 598
   HDAMFSPF keyword 225, 443, 577
   HDLUXASM sample user exit routine 665
   HDPRINT keyword 232, 522
   HDRLEN, USERHDR subkeyword 571

G

GDATE field 683
GDG image copy data sets 330
GLBL command
   Image Copy function 312, 313
   Prefix Resolution/Update function 278
global options module 140
global options modules
   create Index Build function global options module
      using batch job 253
   create Reload function global options module using
      batch job 222
   create Unload function global options module using
      batch job 184
   Image Copy function 318
   Index Build function 251
   Prefix Resolution/Update function 280
   Reload function 219
   Reorg function 101
   specify using ISPF interface
      Index Build function 253
      Reload function 221
      Unload function 183
   Unload function 181
   global services group name keyword 442
   Global Specifications report 596
   GMT date and time field 683
GROUP command 312, 313, 335, 421
GROUP subcommand 168
GRTIME field 683
GSGNAME keyword 442, 576
HDSORT keyword 444, 577
Help, online 31
HIDAM databases
   and USEINDEX keyword 565
   block truncation example 557
   Database Selection facility BEGIN option 193
   Database Selection facility END option 194
   empty database 226
   JCL requirements to build primary and secondary indexes 246
   JCL requirements to build primary index 247
   LIMITS keyword 481
   Reload function API restrictions 240
   Reorg function JCL considerations 96
   search for space in previous block 531
   start each HIDAM root in new block 556
   high performance option keyword 446
High-performance Prefix Resolution function
   JCL example 275
   system flow 272
HIPO keyword 285, 446
HISAM databases
   Database Selection facility BEGIN option 193
   Database Selection facility END option 194
   empty VSAM database 226
   not supported by LIMITS keyword 483
   parallel processing 504
   restrictions for Prefix Resolution/Update function 75
HISAM Unload utility 300
HIST keyword 335, 447
historical analysis, description 38
HLDURIDX DD statement 367, 368
HLDUXASM sample user exit routine 664
HLDUXCOB sample user exit routine 664
HLDUXPLI sample user exit routine 664
HSAM databases
   creating 448
   restrictions 74
   Unload function API 208
HSAMOUT keyword 448
HULUXCOB sample user exit routine 665
HULUXPLI sample user exit routine 665

IC command 312, 313, 349, 389
IC keyword 325, 326, 449
ICACC keyword 331, 449
ICALLOC keyword 326, 450
ICAVGREC keyword 451
ICBUF keyword 332, 451
ICCATF keyword 452
ICDATAACL keyword 453
ICDSCB keyword 454
ICEXPDT keyword 454
ICGDG keyword 330, 455
ICLIKE keyword 455
ICMGMTCL keyword 456
ICNEEDED flag 457, 641, 642, 644, 645
ICNEEDED keyword 457
ICP keyword 458
ICP$DSN 664, 678, 679, 686
ICP$DSN user exit 330
ICP@imsi module
   creating 319, 323
   use 318
ICP@xxxx module
   creating 319, 323
   use 319
ICP@ARMX module
   creating 319, 323
   use of 318
ICPRINT DD statement 374
ICPREF keyword 459
ICPSYSIN
   commands 312
   control statements 311
   DD statement 93, 118, 310
   keywords 313
ICRETPD keyword 460
ICSPACE keyword 461
ICSTORCL keyword 462
ICUNIT keyword 329, 463
ICVOLCNT keyword 463
ICVOLSER keyword 329, 464
IDCAMS ALTER commands data set 351
IDCAMS DELETE/DEFINE statements data set 352
IDCAMS keyword 465
IEESYSAS 543
IEFRDER data set name 483
IEFRDER DD statement 117, 369
ignore free space percentage factor 443
ILDS keyword 468
image copies
   for recovery 107
   guidelines 106
   ICP keyword 458
   restore image copy to original data set 91, 552
image copy data sets
   access method options 331
   block size 333
   buffer pool 332
cataloging 330
DASD 328
image copy data sets (continued)
  data set name 329
  DD statement 310, 325
dynamic allocation 326
  explicit allocation 325
gdgs 330
  identifying to Image Copy function 325
multivolume 328

Image Copy function
  AIC command 349, 389
dynamic allocation 326
  IC command 349, 389
  index databases 77, 309
  invoke with ICP keyword 458
  invoked from Reorg function 107
  JCL requirements 310
  keywords 313, 317, 319
  keywords not valid with IC command 458
  limitations 76, 309
  logical relationships 308
  monitoring options 334
  options 313, 317, 319
  overview 308
  POINTER CHECKER PLUS interface 334
  reporting options 334
  reports 325
  time stamp of image copy data sets 76, 309
  virtual image copies 330
  write options 331

image copy needed keyword 457

IMAGE COPY PLUS 310, 317

IMS DD statement
  Analyzer 151
database scan 190
description 370
  Extractor and Recommendation Selectort 147
  HD Sort utility 230
  online reorganization 155
  Reload function API 236
  Unload function API 203

IMS HD Reorganization Unload utility, differences
  between utility and Unload function 185

IMS HD Scan 301
IMS requirements 70
IMS systems 130
IMS utilities replaced by Index Build function 254
IMS/ESA Year 2000 Exit Tool product 425
IMSACB DD statement, Reload function API 237
IMS-compatible Prefix Resolution function
  JCL example 288
  system flow 286
IMS-compatible records 427
IMSFIRST keyword, Unload function API 204
IMS-format records on SORTIN keyword 544
IMSID keyword 469, 471, 577
IMSID, setting for PDX key 469, 471
INDD keyword 472, 576

INDDX keyword 472
Index Build function
  and DBRC 642
  and Secondary Index Resolution function 300
  BUILD command 389
  BUILDALL command 389
  examples 248
  features and benefits 55
  global options modules 251
  ISPF interface panel map 252
  JCL requirements to build primary and secondary
  indexes 246
  JCL requirements to build primary index 247
  JCL requirements to build secondary indexes 245
  list of reports generated 253
  overview 55
  restrictions 75
  return codes 619
  SIU@imsi module 385
  SIUPARMS module 385
  sort considerations 249
  system flow 244
  virtual image copies 331

index databases
  Image Copy function 77, 309
  Unload function API 208
  virtual image copies 330

Index Load Output Results report 600

INDX command 279
INIT.RECON command 330
initialize empty database 103, 226
initialize empty secondary index 262
input data set
  input data set names and ddnames for Reorg function
  reload 358
  specify input data set for secondary index 472
INPUT keyword 474
INST keyword 476
installation. See the BMC Products for IMS Installation Guide
interrupting and resuming an online reorganization 137
invalid PTF and PTB root pointer chain 564
invoke DBRC 422
IOSTATS keyword 526
IRLMGRP keyword 477

ISPF interface
  build global options module
  Index Build function 253
  Reload function 221
  Unload function 183
  Index Build function 252
  Reload function 221
  Unload function 183
  view reports 584
J

JCL examples
  build primary index 258
  Concurrent Prefix Update 278
create empty secondary index 262
database scan 189
Database Selection facility 193
FRF/EP (logically related databases) 88, 89, 90, 170
HD Sort utility 229
High-performance Prefix Resolution function 275
IMS-compatible Prefix Resolution function 288
IMS-compatible Prefix Resolution function JCL example 288
Index Build function, build primary and secondary indexes 246
Index Build function, build primary index 247
Index Build function, build secondary indexes 245
Prefix Update function 297
Reload function API 235
Reorg function 83
Reorg function, reorganize to image copy 91
Reorg function, using HD Unload file as input 84
restart process using FRFREST0 633
restart process using RESTART command 634
Secondary Index Resolution function 299
Two-step Prefix Resolution function 292
Unload function (parallel unload) 176
Unload function API 202
unload HDAM RBNs 481
unload HIDAM records 482
WF1 Generator 302
Work File Splitter 305

JCL generator 38
JCL requirements
  database scan 189
  online reorganization 154
  Record Analysis 150

K

key list data set 144, 154
See also OLRKEYS DD statement
KEYLEN keyword, HD Sort utility 232
KEYS keyword 526
keyword construction 163
keywords
  See also options
ABNDNO 406
AGN 407
ALLOWLC 407
ALLOWLP 407
AMSOOUT global option 408
AOI 409
AOIUSER 410
AUTHERR 411, 412
BEGIN, Database Selection facility 193
BUFSIZE 414
BUFFSPACE 414
CKUPDATE 414
COMPRESS 225, 415, 419
COUNT, Database Selection facility 193
CREORG 419
DBD 232, 420
DBDSGRP 337, 421
DBR 421
DBRC 422
DBRC (Prefix Resolution/Update function) 423
DDN 423
DESC 242, 490
DESCCODE 232, 491
DFSDBUX1 232, 425
DFSURGU1 187, 425
DYNALLOC 430
END, Database Selection facility 194
EOJACCES 433
EXPAND 187, 415, 417, 419
FTFOHAL 434
FILSZ 437
FIXED 437
FMTRAA 225, 438, 439
FORCELP 439
FREQ 335
FRFPRINT 522
FSPF 441
FUNC 337
GSGNAME 442
HD Sort utility 232
HDAMFSPF 225, 443
HDPRIN 232, 522
HDSORT 444
HIPO 446
HIST 335, 447
HSAMOUT 448
IC 325, 326, 449
ICACC 331, 449
ICACL 330, 449
ICALLOC 326, 450
ICAVGREC 451
ICBUF 332, 451
ICCATF 452
ICDATAACL 453
ICDCSB 454
ICEXPDT 454
ICGDBG 330, 455
ICLIKE 455
ICMGTCL 456
ICNEEDED 457
ICP 458
ICPREF 459
ICPSYSIN 313
ICREATEP 460
ICSPACE 461
ICSTORCL 462
keywords (continued)

ICUNIT 329, 463
ICVOLCNT 463
ICVOLSER 329, 464
IDCAMS 465
ILDS 468
Image Copy function 313, 317, 319
IMSFIRST 204
IMSID 469, 471
INDD 472
INDDX 472
INPUT 474
INST 476
IOSTATS 526
IRLMGRP 477
KEYLEN, HD Sort utility 232
KEYS 526
LDPPRINT 522
LIM 478
LIMITS 479
LOGDSN 483
LOGEXPDT 485
LOGRETPD 485
LOGSPACE 486
LOGUNIT 486
LPCK 187, 487
MINREC 489
MODE 489
MONITOR 233, 490
MONUSERS 233, 490
NAME 336, 493
NBLOCKS 233
NBUF 447
NEWDBD 494
NOAUTH 495
not available with REORG command 575
NOTIFY 495
NRAPS, HD Sort utility 233
OLDDBD 496
Online Reorg function 119
OSAMMAX 500
OUTDDX 501
OVLFONLY 503
PADCHAR 503
PARALLEL 504
FARMBLK 319, 507
PCB
  Database Selection facility 194
  Reload function API 238
  Unload function API 204
PCBNAME
  Database Selection facility 194
  Reload function API 238
  Unload function API 204, 206
PCP 334, 517
PDX 143, 518, 529
PDXPARMS 520
Prefix Resolution/Update function 279, 283
PRIMIND 520
PRINT 194, 204, 206, 522
PRTOLC 521
PRTOLP 521
PSB 523
PRTRCHECK 524
PRTRERROR 524
RAPCHAIN 526
RCDDIST 526
Record Analysis 148
Reload function 216
Reorg function 97
REORMSG 525
REPORTS 526
REPORTS(IOSTATS) 187, 526
REPORTS(KEYS) 187, 526
REPORTS(RAPCHAIN) 526
REPORTS(RCDDIST) 187, 526
REPORTS(ROOTDIST) 526
RMBYTES 528
RMNAME, HD Sort utility 233
ROOTDIST 526
ROUTCODE 490, 530
ROUTCODE 491
SCAN (space search method)
  effect on performance 225
  HDAM 531
  HIDAM 531
SECINDEX 534
SEGPOS 535
SEGPOS 536
SEGSEL 536
SEQERROR 537
SHARE 540
SIUDSN 331, 541
SIUPRINT 522
SIUSORT 542
SIZE 447
SKIP, Database Selection facility 194
SKIPFREQ, Database Selection facility 194
SNAPDBD 204
SNAPERR 205
SNAPSHOT 122, 180
SNPERROR 205
SORTDD 447
SORTIDX 544
SORTIN 544
SORTWORK 233, 545
SPILL 547
SSID 547
STATS 335, 548
summary of keywords 390
SWAP 549
SYNCTAPE 186, 554
syntax 405
keywords (continued)
TIME 556
TRUNC 225, 556
UCBITMAP 557
ULPPRINT 522
UNIT 559
UNLAUTH 559
UNLDD 560
UNLDDLST 562
Unload function 178, 187
UPDATE 564
USEINDEX 564
USEREXIT 225, 567
USERHDR 233, 568
USERID 573
VIC 331, 573
XBMSSID 123, 181, 204, 574
XFIXED 575
xxxPRINT 522

logical relationships and WF1 Generator 303
DFSURWF1 DD statement 364
FRF/EP data set name swapping 629
FRF/EP JCL requirements 88, 89, 90, 170
Image Copy function use with 308
IMS insert rules 224
in HISAM databases 75
information report 601
not supported by CHANGE RECORDING FACILITY 72
PCP keyword 334
PRPURWF1 DD statement 375
Reload function API 236
reorganizing databases 86, ??–90
scan 187
single-step reorganization 87
summary report 601
Logical Twin Resolution
Two-Step Prefix Resolution function 294
use of BMC-format to improve performance 285
with Two-step Prefix Resolution function 292
LOGRETPD keyword 485
logs
data set name 483
expiration date 485
retention period 485
SMS classes 485
space to use for dynamic allocation 486
unit name 486
LOGSMS keyword 485
LOGSPACE keyword 486
LOGUNIT keyword 486
LONG record format 427
LPCK 176
LPCK keyword 187, 487, 577
LREL command 279

main storage size for first sort keyword 447
maintain secondary index 261
maintenance 71
MAXM Reorg/EP Utilities, comparison to existing IMS and BMC utilities 40
MAXM Reorg/Online
overview 37, 48
overview of commands 389
processing phases 64
message log
description 604
online reorganization 160
Recommendation Selector 149
Record Analysis 149, 152
scan mode 525
messages data set 353
migrating to HALDB 689
minimum JCL requirements 156
MINREC keyword 489
minus sign (continuation character) 406
MODE keyword 489
modes, online reorganization 136, 137
See also record mode, scan mode, block mode
MODIFY command 491
monitor job progress 490
MONITOR keyword 233, 490
MONUSERS keyword 233, 490
multiple data set groups, Reorg function JCL
considerations 96
multiple database copies 165
multiple database reorganizations 165
multivolume image copy data sets 328
multivolume OSAM data sets 225
multivolume OSAM databases, restrictions 76
MVS catalog 108, 550
MVS MODIFY command 491
MVS requirements 70
MXE.LOAD library, placement in STEPLIB concatenation
for Reload function API 236
for Unload function API 206

N
NAME keyword 336, 493
NBLOCKS keyword, HD Sort utility 233
NBUF keyword 447
new secondary index, create 260
NEWDBD keyword 494, 576
NOAUTH flag 104, 641, 645
NOAUTH keyword 495
NOCATDS parameter in DBRC 330
NOFEOV keyword setting 421
NONRECOV databases
and DBRC 643
ICNEEDED keyword 457
notification to DBRC 103, 125, 644
NOTIFY keyword 495
notify user IDs keyword 495
NOTIFY.IC command 331
NOTIFY.UIC, and recovery 642
NRAPS keyword, HD Sort utility 233
number of chain buffers 447

O
OLDDBD keyword 418, 496
OLRKEYS DD statement 155
See also key list data set
See also KEYS keyword
online Help 31
online IMS systems 130

Online Reorg function
AOI 126
examples 123
features and benefits 48
FRF$GLBL module 385
Image Copy function use 308
keywords 119
list of reports generated 124
overview 109
post-processing phase 125
REORG command 389
restart post-processing tasks 631
restrictions 72
system flow 109
task overview 113
online reorganization
examples 156
free space determination 136
interrupting and resuming 137
JCL requirements 154
keywords 159
modes 136, 137
OREORG command 141
reports 159
online reorganization overview 48
options
See also keywords
construction 163
DLIMPARM 140
global 140
limit number of report sets saved in PDX 584
online reorganization 159
Record Analysis 152
TRACE 622
Options in Effect report 604
options in effect report 160
OREORG command 141
OS/390 requirements 70
OS/VS COBOL user exit 677
OSAM
allocate multivolume OSAM data sets 225
block updates 588
database larger than 4 GB 500
Reload function 236
restrictions for multivolume databases 76
OSAMMAX keyword 500, 577
OUTDDDX keyword 501
output data set
create customized user output data set 568
messages 353
output data set names and ddnames for Reorg
function 81
reports 354
secondary output data set for unloaded data 360
unloaded data 360, 361, 362, 363
Output Data Set Summary report 605
overflow area, place segments in HDAM overflow area 503
OVLFONLY keyword 503

P
PADCHAR keyword 503
padding segments 503
PARALLEL keyword 504
Parallel Unload facility 479
parameters. See keywords
Parent/Child Segment Statistics report 606
Parent-Child Resolution, use of BMC-format records to improve performance 285
PARMBLK keyword 319, 507
Partition Stats report 606
PCB keyword
  Database Selection facility 194
  Reload function API 238
  Unload function API 204
PCBNAME keyword
  Database Selection facility 194
  Reload function API 238
  Unload function API 204, 206
PCP keyword 334, 517
PCPDANIP DD statement 151
PCPDO001 DD statement 147
PCPDROOT DD statement 147
PCPSYSIN DD statement 148, 334
PDX
  bypass usage 371
  Image Copy function 335
  Image Copy function options 319
  limit number of reports saved 585
  specify DBD name portion of PDX key 520
PDX DD statement 156, 191, 371
PDX key, IMSID portion 469, 471
PDX keyword 143, 518, 529, 577
PDXDSN 143, 156
PDXPARMS keyword 520, 577
performance considerations
  change HDAM randomizer 224
  Concurrent Prefix Update 564
  fixed length sorting 575
  number of chain buffers 447
  Prefix Resolution/Update function 285
  Reload function 224
  Reorg function and image copy restore 554
  Unload function 186
PHDAM, see HALDB
PHIDAM, see HALDB
physical pointers 524
place segments in HDAM overflow area 503
plus sign (continuation character) 406
PLUSDUMP DD statement 311, 620
PLUSIN control statement data set
  Database Selection facility 193
description 372
  HD Sort utility 232
  Index Build function 248
  Online Reorg function 123
  Reload function 215
  Reload function API 237
  Reorg function 93
  syntax 404
  Unload function 177
  Unload function API 203
PLUSIN DD statement 148, 152, 155, 191, 372
PLUSLIST DD statement 373, 620
PLUSOUT DD statement 311, 373, 621
POINTER CHECKER PLUS 334
pointer errors continue after pointer error 524
  PTRERROR keyword 524
pointers
  Bad Block/Pointer report 585
  check physical pointers 524
  POS=M, Unload function API 208
post-processing phase complete tasks manually 630
description 103, 125, 627
restart 631
prefix of HD Unload records 426
prefix resolution and update 52, 241
Prefix Resolution/Update function
  batch keywords 283
  GLBL command 278
  global options modules 280
  High-performance Prefix Resolution function 272
  IMS-compatible Prefix Resolution 286
  INDX command 279
  keywords 279
  list of reports generated 284
  LREL command 279
  performance considerations 285
  Prefix Update 295
  PRP@imsi module 385
  PRPPARMS module 385
  return codes 619
  Two-step Prefix Resolution 289
  WF1 Split function 241
  Work File Splitter 304
Prefix Update function
  description 295
  JCL example 297
  system flow 295
prefix update keyword 564
primary database
  Reorg function JCL considerations 94
  specify DBD name 420
primary index databases, virtual image copies 331
primary index option 520
PRIMIND keyword 520, 576
PRINT DD statement 374
PRINT keyword
  Database Selection facility 194
description 522
  Unload function API 204, 206
print logical child orphans keyword 521
print logical parent orphans keyword 521
Process Summary report 607
Processing Log report 607, 608
processing phases of MAXM Reorg/Online 64
product changes 32
product support 3
PRP@imsi 280
PRP@imsi module 385
PRPDSEH0 (WF1 Generator) 300
PRPPARMS 280, 283
PRPPARMS module 385
PRPSPLIT (Work File Splitter) 304
PRPURG10 (High-performance Prefix Resolution function) 274
PRPURG20 (Parent-Child Resolution) 274, 291, 293
PRPURG30 (Logical Twin Resolution) 274, 292, 294
PRPURG40 (Secondary Index Resolution) 285, 291, 298
PRPURGP0 (Stand-alone Prefix Update function) 295
PRPURIDX DD statement 375
PRPURIDX work data set 559
PRPURWF1 data set 213
PRPURWF1 DD statement 191, 375
PRTOLC keyword 521
PRTOLP keyword 521
PSB conditions under which Unload function API works 208
PSB keyword 523
PTF and PTB broken/invalid root pointer chain 564
PTRCHECK keyword 524, 576
PTRERROR keyword 524, 576
publications, related 31

R
RAA, format complete RAA 438, 439
randomizer changes 74, 224, 227, 497
RAP Sequencer 444
RAPCHAIN keyword 526
RCDDIST keyword 526
RCUDSN 679
read access to database during reorganization 540
read exclusive access to database during reorganization 541
rebuild existing secondary index 261
recommendation report 149, 152
Recommendation Selector 145, 146
RECON data sets 644
RECONx DD statement 155, 376

Record Analysis
Analyzer 144, 150
commands 141
Extractor 144
JCL requirements 150
reports 149, 152
record format
FRMT1 428
FRMT2 429
FRMT3 429
LONG 427
SHORT 427
XSHORT 427
record mode 137, 156
record prefix, selecting format of HD Unload records 426
RECOV.NEEDED flag 641, 645
recovering a database reorganized by the Online Reorg function 133
RECOVERY PLUS for IMS 76, 309
region controller
  Reload function 212
  Unload function 174
register usage
  Reload function user exit routine 669
  Unload function user exit routine 666
related publications 31
RELOAD command 389
  See also Reload function
reload database in parallel mode 504
Reload function
  See also Reload function API
  DBUDSEH1 module 301
differences between Reload function and IMS HD Reorganization Reload utility 223
examples 215
features and benefits 54
global options modules 219
ISPF interface panel map 221
keywords 216, 225
LDP@imsi module 385
LDPPARMS module 385
list of reports generated 222
overview 54
performance 224
RELOAD command 389
restrictions 74
return codes 619
SORT command 389
system flow 212
Reload function API
description 234
DL/I status codes 239
invalid keywords 237
JCL example 235
restrictions 239
system flow 234
USERLOAD command 389
Index 735

A   B   C   D   E   F   G   H   I   J   K   L   M   N   O   P   Q   R   S   T   U   V   W   X   Y   Z

remote site recovery option 442
REORG command 389
   See also Reorg function
Reorg function
   and DBRC 643
   AOI 409
data set name swapping 51, 549
   examples 93
features and benefits 49
FRFSGLBL module 385
global options module 101
Image Copy function use 308
input data set names and ddnames 81
JCL considerations 94
JCL requirements 83
JCL requirements, reorganize to image copy 91
JCL requirements, using HD Unload file as input 84
   keywords 97
output data set names and ddnames 81
   overview 49
post-processing phase 103
REORG command 389
   reports 102
   restart post-processing tasks 631
   restrictions 73
   system flow 80
   task overview 82
unload file used as input 474
reorganization for logically related databases 86
reorganize database in parallel mode 504
REORMSG keyword 525
replicate a database 108, 550
reports
   Bad Block/Pointer 585
   banner page 160
   Concurrent Reorganization 586
   Data Set Buffer Statistics 587
   Data Set Group Summary 588
data set name 159
   Database Selection facility output printed format 197
   Database Summary 586
   Dataset List 591
   Dataset Statistics 591
   Datasets Used 592
   DBD Change Summary 592
   DBD Summary 592
   Disassembled DBD 594
   Distribution of Database Record Sizes 594
   Distribution of Free Space 595
   Distribution of HDAM RAP Chain Lengths 595
extract statistics 581
generated by Index Build function 253
generated by Online Reorg function 124
generated by Prefix Resolution/Update function 284
generated by Reload function 222
generated by Reorg function 102
generated by Unload function 184
Global Specifications 596
HD Sort - Record Counts 599
HDAM Root Addressable Area Statistics 597
HDAM Root Placement Relative to Its Home Block 526, 598
Image Copy function 325
Index Load Output Results 600
Largest Database Records 526, 600
limit number of report sets saved in PDX 584
Listing of PLUSIN Control Statements 601
Logical Relationship Information 601
Logical Relationship Summary 601
message log 604
multiple database reorganizations/copies 170
online reorganization 159
Options in Effect 604
Output Data Set Summary 605
Output Image Copy Summary 605
Parent/Child Segment Statistics 606
Partition Stats 606
Process Summary 607
Processing Log 607, 608
Recommendation Selector 149
Record Analysis 149, 152
Secondary Index Definition 609
Secondary Index Relationship Summary 611
Segment I/O Requirements Matrix 612
Segment Statistics 612
Status of Segments Read from HD Unload Dataset 613
summary of reports 582
Twin Segment Statistics 614
User Exit Routine Action 614
Variable Length Segments Statistics 615
view using ISPF interface 584
VL Segment Split Statistics 616
reports data set 354, 522
REPORTS DD statement 311, 581
REPORTS keyword 526, 577
REPORTS(IOSTATS) keyword 187, 526
REPORTS(KEYS) keyword 187, 526
REPORTS(RAPCHAIN) keyword 526
REPORTS(RCDDIST) keyword 187, 526
restart information data set 408
restart process
   description 631
   JCL example using FRFREST0 633
   JCL example using RESTART command 634
Restart Swap utility. See restart process
restore image copy to original data set 91, 552
restrictions
   Index Build 75
   MAXM Reorg solutions 72
   Online Reorg function 72
   Reload function 74
   Reload function API 239

Index  735
restrictions (continued)

Reorg function 73
Unload function 74
Unload function API 207
restructure database 494
retention period for logs 485
return codes
  MAXM Reorg solutions 617
    Online Reorg function user exit routine 676
    Reload function user exit routine 670
    Unload function user exit routine 667
REUSE option 95, 553
RMBYTES keyword 528, 577
RMNAME keyword, HD Sort utility 233
root addressable area, format complete RAA 438, 439
root segment, placement of segments 535
ROOTDIST keyword 526
ROOTKEY, USERHDR subkeyword 571
ROOTRBA, USERHDR subkeyword 571
ROUTCODE keyword 490, 530
ROUTCODE keyword 491
routing codes 491

secondary index records
  and WF1 Generator 303
  sorting 544
Secondary Index Relationship Summary report 611
Secondary Index Resolution function
description 298
  JCL example 299
  system flow 298
SEGKEY, USERHDR subkeyword 571
SEGKEYOF, USERHDR subkeyword 571
SEGKEYSZ, USERHDR subkeyword 571
SEGLEV1, USERHDR subkeyword 571
SEGLEV2, USERHDR subkeyword 571
Segment I/O Requirements Matrix report 526, 612
  segment position option 535
  Segment Statistics report 612
segments
  padding 503
  place segment near root 535
  unload selected segments 536
  unloading and reloading compressed segments 416, 419
SEGNAME, USERHDR subkeyword 571
SEGPOS keyword 535, 577
SEGPOSO keyword 536
SEGSEL keyword 536, 576
setups 405
SEQUERROR data set 539
SEQUERROR DD statement 377
SEQUERROR keyword 537, 576
sequence error 537
  sequence error data set 377, 539
  sequential access method 331
setting options using ISPF interface
  Index Build function 253
  Reload function 221
  Unload function 183
setup member 647
See also the BMC Products for IMS Installation Guide
SHARE keyword 540
shared secondary index support 331, 574
shared secondary indexes 73, 75, 95, 553
SHARELVL 640
SHSAM databases
  empty VSAM database 226
  parallel processing 504
SHORT record format 427
SHSAM databases restrictions 74
signon failures for DBRC 618
SIU@imsi module 251, 385
SIU@SN keyword 331, 541

SCAN (HDAM) keyword 531
SCAN (HIDAM) keyword 531
SCAN (space search method) keyword
effect on performance 225
  with REORG command 577
scan database to build index 257
scan function 187
scan mode 139, 158
search for space in previous block 531
search techniques 532
SECINDEX keyword 534, 576
secondary index
  See also shared secondary indexes
    add during reorganization 498
    and Reorg function 105
    create empty secondary index 262
    create new secondary index 260
    delete during reorganization 498
    maintain 261
    read secondary index records from separate files 472
    rebuild existing secondary index 261
    Reload function API 236
    Reorg function JCL considerations 95
    root retrieval 565
    scan primary database to build secondary index 257
    secondary index keyword 534
    summary report 611
    USEINDEX keyword 566
    virtual image copies 331
    write secondary index records to separate files 501
Secondary Index Definition report 609

SIGNON keyword 538
Silent Run function 657
SIUPARMS 251
SIUPARMS module 385
SIUPRINT DD statement 374
SIUPRINT keyword 522, 577
SIUSORT keyword 542
SIZE keyword 447
SKIP keyword, Database Selection facility 194
SKIPFREQ keyword, Database Selection facility 194
SMS image copy data sets 329
SNAPDBD keyword 204
SNAPERR keyword 205
Snapshot Copy 204, 205
Snapshot Copy function 67
SNAPSHOT keyword 122, 180
SNPERROR keyword 205
SORT command 232, 389
sort considerations for the Index Build function 249
sort function 227
sort index records keyword 544
sort indexes in multiple address spaces 542
sort keyword 444
sort product requirements 71
sort work data set 545
SORTDD keyword 447
SORTIDX keyword 285, 544
SORTIN DD statement 377
SORTIN DD statement, HD Sort utility 231
SORTIN keyword 377, 544
sorting index records 544
SORTOUT DD statement, HD Sort utility 231
SORTWK DD statement 378
SORTWKnn DD statement, HD Sort utility 232
SORTWORK keyword 233, 545, 577
space search method
  flip-flop technique 533
  HDAM 531
  HIDAM 531
specifying options by using ISPF interface
  Index Build function 253
  Reload function 221
  Unload function 183
SPILL keyword 547
SSID keyword 547
stacked image copies not supported 76, 309
start each HIDAM root in new block 556
statistics, Image Copy function 335
STATS keyword 335, 548
status codes
  Reload function API 239
  Unload function API 207
Status of Segments Read from HD Unload Dataset report 613
STEPLIB DD statement
  Analyzer 150
description 378
  Extractor and Recommendation Selector 146
  HD Sort utility 230
  HD Unload function 190
  online reorganization 155
  Reload function API 236
  Unload function API 203
substitution characters for ddnames 560
summary of changes 32
summary of keywords 390
super-abbreviated records 427
support, customer 3
suspend-and-resume interface 114
SWAP keyword 549
symbolic pointers 187
synchronize tape volumes 554
SYNCTAPE keyword 186, 554, 576
syntax
  commands 405
  comments 405
  continuation characters 406
  keywords 405
  separators 405
SYSOUT DD statement 232, 379
SYSPRINT DD statement
  data set for reports 581
database scan 190
  HD Sort utility 231
system flow
  database scan 189
  HD Sort utility 229
  Index Build function 244
  Online Reorg function 109
  Online/Defrag record analysis 144
  Online/Defrag reorganization 153
  Reload function 212
  Reload function API 234
  Reorg function 80
  Unload function 174
  Unload function API 200
system requirements 70
SYSUDUMP DD statement 147, 151, 191, 380, 621
T
  tape
    image copy data sets 327
    synchronize unload volumes 554
task overview
  Online Reorg function 113
  Reorg function 82
technical support 3
TIME keyword 556
time stamp of output image copies 76, 309
TRACE command 205, 389
trace DD statement 620, 621
  Trace facility keywords 623
  TRACE keyword 622
tracing work file records 622
TRUNC keyword 225, 556, 577
truncation example 557
Twin Segment Statistics report 614
twin segments
  continue after sequence error 537
  sequence errors 539
Two-step Prefix Resolution function
description 289
  JCL example 292
  system flow 290

U
U0041 abend code 618
U0047 abend code 618
UCBITMAP keyword 557
ULP@imsi module 181, 385
ULPPARMS module 181, 385
ULPPRINT DD statement 374
ULPPRINT keyword 522, 577
unabbreviated records 427
UNIT keyword 559
unit name for dynamically allocated work data sets 559
UNLAUTH keyword 559
UNLD keyword 560
UNLDDLST keyword 562
Unload Authorization keyword 559
UNLOAD command 206, 389
  See also Unload function
unload database in parallel mode 504
unload file as input to Reorg function 84
unload file used for Reorg function reorganization 474
unload file, used as input to Reorg function 475
Unload function
differences between Unload function and IMS HD
  Reorganization Unload utility 185
  examples 177
features and benefits 52
global options modules 181
ISPF interface panel map 183
JCL requirements for parallel unload 176
  keywords 178, 187
list of reports generated 184
overview 52
performance 186
restrictions 74
return codes 618
system flow 174
TRACE command 389
ULP@imsi module 385
ULPPARMS module 385
UNLOAD command 389
USERREAD command 389
with Snapshot Copy 67
Unload function API
  control statement data set 203
  description 199
  DL/I conditions under which API works 207
  DL/I status codes 207
  JCL example 202
  PSB conditions under which API works 208
  restrictions 207
  system flow 200
Unload function COBOL example 665
unload in parallel 479
UNLOAD PLUS/EP API 204, 205
unload record format 425
unload selected segments 536
unused files in Reorg function 97
update access to database during reorganization 540
UPDATE keyword 564
UPDATE option
  improving performance 285
  with Concurrent Prefix Update 277
USEINDEX keyword 564, 566, 576
user exit routine
  and fixed-length segments 665
  AOI 410
  COBOL considerations 677
  COBOL II 677
  COBOL/370 678
  compressed data 665
  Data Set Name 330, 664, 678
  DFSDBUX1 425
  examples 664
  invoke 567
  linkage section 665
  Online Reorg function Assembler example 664
  Online Reorg function COBOL example 664
  Online Reorg function return codes 676
  OS/VS COBOL 677
  Reload function 669
  Reload function Assembler example 664
  Reload function COBOL example 664
  Reload function PLI example 664
  Reload function register usage 669
  Reload function return codes 670
  Unload function 667, 675
  Unload function Assembler example 665
  Unload function COBOL example 665
  Unload function register usage 666
  Unload function return codes 667
  User Exit Routine Action report 614
User Exit Routine Action report 614
User Format 1 records 428
User Format 2 records 429
User Format 3 records 429
user header record
  specify format of 568
  subkeywords 571
user load program. See Reload function API
USEREXIT keyword 225, 480, 567, 577
USERHDR keyword
description 568
example 572
HD Sort utility 233
not available with REORG command 576
subkeywords 571
USERID keyword 573
USERLOAD command 237, 389
USERREAD command 204, 389

V
Variable Length Segment Split Statistics report 616
Variable Length Segment Statistics report 615
VIC keyword 331, 573
virtual image copies
DBDSGRP keyword 421
index database 330
with SIUDSN keyword 542
VL Segment Split Statistics report 616
VSAM
control interval updates 588
KSDS buffers 186
REUSE option 95
VSAM REUSE option 553

W
WF1 data set 364
WF1 Generator
description 300
JCL example 302
system flow 301
work file 1 data set 364
work file records, compatibility of BMC format and IMS format 300
Work File Splitter
description 304
JCL example 305
system flow 304

X
XBM for IMS 204, 205, 574
XBM Snapshot for IMS 543
XBMSSID keyword 123, 181, 204, 574
XFIXED keyword 575
XIM 167, 695
XIMCHECK 702
XSHORT record format 427
xxxPRINT DD statement 374
xxxPRINT keyword 522

Y
year 2000 exit 425